Linux Transfer for Windows Network Admins:

A roadmap for building a Linux file and print server



Michael Jang

Edited by
Elizabeth Zinkann

Hentzenwerke Publishing

Linux Transfer for Windows Network Admins: A Roadmap for Building a Linux File and Print Server

Michael Jang

Hentzenwerke Publishing

Published by:
Hentzenwerke Publishing
980 East Circle Drive
Whitefish Bay WI 53217 USA

Hentzenwerke Publishing books are available through booksellers and directly from the publisher. Contact Hentzenwerke Publishing at:
414.332.9876
414.332.9463 (fax)
www.hentzenwerke.com
books@hentzenwerke.com

Linux Transfer for Windows Network Admins: A Roadmap for Building a Linux File and Print Server
By Michael Jang
Technical Editor: Elizabeth Zinkann
Copy Editor: Jeana Frazier
Cover Art: "Network" by Todd Gnacinski, Milwaukee, WI

Copyright © 2003 by Michael Jang

All other products and services identified throughout this book are trademarks or registered trademarks of their respective companies. They are used throughout this book in editorial fashion only and for the benefit of such companies. No such uses, or the use of any trade name, is intended to convey endorsement or other affiliation with this book.

All rights reserved. No part of this book, or the ebook files available by download from Hentzenwerke Publishing, may be reproduced or transmitted in any form or by any means, electronic, mechanical photocopying, recording, or otherwise, without the prior written permission of the publisher, except that program listings and sample code files may be entered, stored and executed in a computer system.

The information and material contained in this book are provided "as is," without warranty of any kind, express or implied, including without limitation any warranty concerning the accuracy, adequacy, or completeness of such information or material or the results to be obtained from using such information or material. Neither Hentzenwerke Publishing nor the authors or editors shall be responsible for any claims attributable to errors, omissions, or other inaccuracies in the information or material contained in this book. In no event shall Hentzenwerke Publishing or the authors or editors be liable for direct, indirect, special, incidental, or consequential damages arising out of the use of such information or material.

ISBN: 1-930919-46-8

Manufactured in the United States of America.

Our Contract with You, The Reader

In which we, the folks who make up Hentzenwerke Publishing, describe what you, the reader, can expect from this book and from us.

Hi there!

I've been writing professionally (in other words, eventually getting a paycheck for my scribbles) since 1974, and writing about software development since 1992. As an author, I've worked with a half-dozen different publishers and corresponded with thousands of readers over the years. As a software developer and all-around geek, I've also acquired a library of more than 100 computer and software-related books.

Thus, when I donned the publisher's cap five years ago to produce the *1997 Developer's Guide,* I had some pretty good ideas of what I liked (and didn't like) from publishers, what readers liked and didn't like, and what I, as a reader, liked and didn't like.

Now, with our new titles for 2003, we're entering our sixth season. (For those who are keeping track, the '97 DevGuide was our first, albeit abbreviated, season, the batch of six "Essentials" for Visual FoxPro 6.0 in 1999 was our second, and, in keeping with the sports analogy, the books we published in 2000 through 2003 comprised our third and subsequent seasons.)

John Wooden, the famed UCLA basketball coach, posited that teams aren't consistent; they're always getting better—or worse. We'd like to get better…

One of my goals for this season is to build a closer relationship with you, the reader. In order for us to do this, you've got to know what you should expect from us.

- You have the right to expect that your order will be processed quickly and correctly, and that your book will be delivered to you in new condition.
- You have the right to expect that the content of your book is technically accurate and up-to-date, that the explanations are clear, and that the layout is easy to read and follow without a lot of fluff or nonsense.
- You have the right to expect access to source code, errata, FAQs, and other information that's relevant to the book via our Web site.
- You have the right to expect an electronic version of your printed book to be available via our Web site.
- You have the right to expect that, if you report errors to us, your report will be responded to promptly, and that the appropriate notice will be included in the errata and/or FAQs for the book.

Naturally, there are some limits that we bump up against. There are humans involved, and they make mistakes. A book of 500 pages contains, on average, 150,000 words and several megabytes of source code. It's not possible to edit and re-edit multiple times to catch every last

misspelling and typo, nor is it possible to test the source code on every permutation of development environment and operating system—and still price the book affordably.

Once printed, bindings break, ink gets smeared, signatures get missed during binding. On the delivery side, Web sites go down, packages get lost in the mail.

Nonetheless, we'll make our best effort to correct these problems—once you let us know about them.

In return, when you have a question or run into a problem, we ask that you first consult the errata and/or FAQs for your book on our Web site. If you don't find the answer there, please e-mail us at **books@hentzenwerke.com** with as much information and detail as possible, including 1) the steps to reproduce the problem, 2) what happened, and 3) what you expected to happen, together with 4) any other relevant information.

I'd like to stress that we need you to communicate questions and problems clearly. For example…

- "Your downloads don't work" isn't enough information for us to help you. "I get a 404 error when I click on the **Download Source Code** link on **www.hentzenwerke.com/book/downloads.html**" is something we can help you with.
- "The code in Chapter 10 caused an error" again isn't enough information. "I performed the following steps to run the source code program DisplayTest.PRG in Chapter 10, and I received an error that said 'Variable m.liCounter not found'" is something we can help you with.

We'll do our best to get back to you within a couple of days, either with an answer or at least an acknowledgment that we've received your inquiry and that we're working on it.

On behalf of the authors, technical editors, copy editors, layout artists, graphical artists, indexers, and all the other folks who have worked to put this book in your hands, I'd like to thank you for purchasing this book, and I hope that it will prove to be a valuable addition to your technical library. Please let us know what you think about this book—we're looking forward to hearing from you.

As Groucho Marx once observed, "Outside of a dog, a book is a man's best friend. Inside of a dog, it's too dark to read."

Whil Hentzen
Hentzenwerke Publishing
September 2003

List of Chapters

Introduction	xix
Chapter 1: Basic Linux Installation	1
Chapter 2: Installing Linux as a File Server	29
Chapter 3: Setting Up Your Server File System	59
Chapter 4: Setting Up Your File Server's Users	105
Chapter 5: Connecting Linux Workstations	135
Chapter 6: Connecting Windows Workstations	153
Chapter 7: Configuring Printers	199
Chapter 8: Administration and Management	229
Chapter 9: System Backup	265
Appendix A: Samba 3.0 Preview	287
Appendix B: Sample Samba Configuration Files	291

Table of Contents

Our Contract with You, The Reader iii
Acknowledgements xiii
About the Authors xv
How to Download the Files xvii

Introduction xix

Chapter 1: Basic Linux Installation 1
- Basic file server functions 1
- Selecting a distribution 1
 - Red Hat Linux 3
 - United Linux 3
 - Other Linux distributions 4
- A brief guide to installing Linux 4
 - 1. Checking hardware 5
 - 2. Planning partitions 8
 - 3. Understanding the Filesystem Hierarchy Standard 9
 - 4. Selecting a filesystem format 11
 - 5. Preparing partitions and hard drives for dual booting 15
 - 6. Basic steps to installing Red Hat Linux 24
- Conclusion 27

Chapter 2: Installing Linux as a File Server 29
- Basic terms 29
- The installation nitty-gritty 30
 - Remaining steps 30
 - Configuring a firewall 31
 - Configuring communication through the firewall 32
 - Authentication 33
 - Samba authentication 34
- Installing what's necessary 35
 - Basic components 35
 - Basic components plus 35
 - Options 35
 - Other network services 39
- After installation 40
 - First Boot 40
 - The Red Hat Linux GUI 43
 - The Red Hat Network 44

Rawhide 48
Other updates 48
Package management 48
Installing and upgrading RPMs 48
Using the rpm command 50
rpm command details 51
Uninstalling what is not necessary 52
Checking installed services 53
Getting more information 55
Uninstalling a service 56
Deactivating a service 56
Conclusion 58

Chapter 3: Setting Up Your Server File System 59
The Microsoft CIFS 60
Background 60
Basic look and feel 61
The NFS alternative—an overview 63
Basic administration 63
Process management 64
User categories 65
Configuring a Samba server 66
Getting Samba 66
Samba configuration files 67
Samba and firewalls 68
The Red Hat Samba Configurator 70
Samba Web Administration Tool 75
Configuring Samba in detail 84
Opening smb.conf in a GUI 85
Global settings 86
Defining shared directories and printers 94
Conclusion 102

Chapter 4: Setting Up Your File Server's Users 105
Users and accounts 106
Linux user and group accounts 106
Configuring Samba users 111
Users on a Domain 114
Using the Microsoft user database 114
Setting up a Samba PDC database 118
File and directory management 121
Linux permissions 121
Default permissions 122
Linux file ownership 123
Limited support for Access Control Lists 123

Linux groups 123
Red Hat Linux's private groups 123
Creating a special group 124
Quotas 127
The boot process 127
Configuring quota configuration files 128
Configuring quotas for a user 129
Configuring quotas for a group 131
Setting a grace period 132
Activating quotas 132
Conclusion 133

Chapter 5: Connecting Linux Workstations 135
Configuring the workstation 136
Samba client packages 136
Connecting to a Domain 137
Finding shared directories 137
Mounting directories 139
Linux login batch files 143
Peer-to-peer Workgroups 147
Setting up accounts 149
A PDC and a Windows Domain member server 149
Windows PDC and Linux Domain member server 150
Linux PDC and Linux Domain member server 151
Conclusion 152

Chapter 6: Connecting Windows Workstations 153
Preparing accounts 153
Logon scripts 154
Profiles 154
Configuring the Microsoft workstation 155
Connecting a Windows 95/98/ME workstation to a Domain 156
Creating a Windows 95/98/ME Workgroup share 167
Windows NT 4 Workstation 168
Windows 2000 Professional 177
Windows XP Professional 186
Text-mode network commands 193
Troubleshooting 195
Samba syntax 195
Samba logs 196
Conclusion 197

Chapter 7: Configuring Printers 199

Packages 199
CUPS packages 200
LPD packages 201
Configuring a local printer 201
Checking services 201
Configuring a print service 202
Configuring printers with the CUPS Web-based tool 206
Network printers 211
Creating network printers 212
Sharing printers over a network 214
Connecting to network printers from workstations 215
Connecting to a Linux print server from a Windows client 215
Connecting to a Windows print server from a Windows client 221
Connecting to a Linux print server from a Linux client 222
Connecting to a Windows print server from a Linux client 224
Conclusion 227

Chapter 8: Administration and Management 229

Shells 229
Basic navigational commands 230
Basic file management commands 235
Text editing 241
Scripts for repetitive tasks 243
The cron scheduler 243
cron rotates logs 245
cron helps find files 246
A one-time job 247
Administrative tools 248
Managing the installation files 248
Managing the Linux boot process 250
The Red Hat Linux rescue mode 257
Regular user tasks 260
Conclusion 263

Chapter 9: System Backup 265

Backup strategies and types 265
Backup strategies 266
Backup types 266
RAID 267
Backup options 270
Media options 270
System backups 271

Backup and Restore commands 272
tar archives 273
cpio archives 274
Full, incremental, and differential backups 275
Recording to CDs and DVDs 276
Backing up over a network 280
Backing up from a Windows client to a Windows file server 280
Backing up from a Windows client to a Linux file server 281
Backing up from a Linux client 282
Scheduled backups 283
Creating a file server script 283
Other commands 284
Conclusion 284

Appendix A: Samba 3.0 Preview 287

Appendix B: Sample Samba Configuration Files 291

Acknowledgements

The world of Linux includes a number of communities, with people of all personalities and motives. For example, some of the communities are working on elements such as the kernel, the GUI, and print servers. Together, they've developed an operating system which many find to be superior to the Microsoft server operating systems. With Samba, you can substitute a Linux computer directly for a Microsoft server.

It's taken a small community to create this computer book. With the *Linux Transfer* series, I thank Whil Hentzen for leading the charge to help Microsoft Windows administrators and developers adapt to Linux. Elizabeth Zinkann is a marvelous technical editor, superb Linux administrator, and great friend. As a copy and project editor, Jeana Frazier has done a marvelous job keeping the book on track. Lori Lathrop has developed a great index for this book.

On a personal note, the struggle through widowhood is not easy. It is one day—sometimes one minute—at a time. It would have been much more difficult without Elizabeth's help. She gave me the emotional support I needed in those earliest dark days. It would have been much more difficult without the communities of younger widows and widowers that I've found, both nearby and online.

It is possible to find love again. Donna, as we have shared our journeys through widowhood, we have grown to love each other. I honor your dearly departed Randy as you honor my dearly departed Nancy. I love you completely.

Nancy, I miss you. I thank you for the wisdom that you've given me. I thank you for helping to make me into the person that I am now. You will always be a part of me. I will always love you.

I hope; therefore, I can live.

—*Michael Jang*

About the Authors

Michael Jang

Michael Jang (RHCE, LCP, Linux+, MCSE) is currently a full-time writer, specializing in operating systems and networks. His experience with computers goes back to the days of jumbled punch cards. He has written a number of other books on Linux and Linux certification, including *Mastering Red Hat Linux 9*, *Red Hat Certified Engineer Study Guide*, *Linux+ Exam Cram*, and *Sair GNU/Linux Installation and Configuration Exam Cram.* His other Linux books include *Mastering Linux, Second Edition* and *Linux Networking Clearly Explained.* He has also written or contributed to books on Microsoft operating systems, including *MCSE Guide to Microsoft Windows 98*, *Windows 98 Exam Prep*, and *Mastering Windows XP Professional, Second Edition.*

Elizabeth Zinkann

Elizabeth Zinkann is a logical Linux catalyst, a freelance technical editor, and an independent computer consultant. She was a contributing editor and review columnist for *Sys Admin Magazine* for 10 years. (Her most recent reviews have taken refuge at **www.equillink.com**.) Her articles have also appeared in *Performance Computing*, *Linux Magazine*, and *Network Administrator* magazines. As an independent computer consultant, she has built Linux servers, maintained Linux, Solaris, Macintosh, and Windows computers, programmed databases, and taught Linux, Unix, computer hardware basics, and Internet essentials. In a former life, she also programmed communications features, including ISDN, at AT&T Network Systems.

How to Download the Files

Hentzenwerke Publishing generally provides two sets of files to accompany its books. The first is the source code referenced throughout the text. Note that some books do not have source code; in those cases, a placeholder file is provided in lieu of the source code in order to alert you of the fact. The second is the e-book version (or versions) of the book. Depending on the book, we provide e-books in either the compiled HTML Help (.CHM) format, Adobe Acrobat (.PDF) format, or both. Here's how to get them.

Both the source code and e-book file(s) are available for download from the Hentzenwerke Web site. In order to obtain them, follow these instructions:

1. Point your Web browser to **www.hentzenwerke.com**.
2. Look for the link that says "Download."
3. A page describing the download process will appear. This page has two sections:
 - **Section 1:** If you were issued a user name/password directly from Hentzenwerke Publishing, you can enter them into this page.
 - **Section 2:** If you did not receive a user name/password from Hentzenwerke Publishing, don't worry! Just enter your e-mail alias and look for the question about your book. Note that you'll need your physical book when you answer the question.
4. A page that lists the hyperlinks for the appropriate downloads will appear.

Note that the e-book file(s) are covered by the same copyright laws as the printed book. Reproduction and/or distribution of these files is against the law.

If you have questions or problems, the fastest way to get a response is to e-mail us at **books@hentzenwerke.com**.

Introduction

As Microsoft licensing fees increase, Microsoft administrators are looking for alternatives. In the world of servers, the main alternatives to Microsoft Windows are those operating systems related to Unix. With Samba, Unix-style operating systems can take their place as a workstation or server on a Microsoft-based network. Linux stands out because it is freely available. Red Hat Linux is by far the most popular Linux distribution.

This book is targeted at Microsoft Windows administrators. As an author with an MCSE and an RHCE, I think I know how to describe how to use Linux with the language of the Microsoft network administrator. In this book, I assume that the reader is fairly new to Linux. Die-hard Linux administrators may be disappointed; I do not cover the elements of Samba that would be too difficult for the average Microsoft Windows administrator to handle.

With a few easy tweaks, you can set up a Linux computer on a Microsoft network. You can set it up as a workstation or a server. You can set up Linux as a member or a Primary Domain Controller (PDC) on a Microsoft-style Domain. This book illustrates how to do this with Red Hat Linux 9, which you can easily download from various Internet sites. Red Hat Linux 9 is the last freely available Linux distribution developed and packaged by Red Hat.

Red Hat Linux 9 comes with Samba 2.2.7, and this book is based on that software. If you choose to upgrade to Samba 3.0, I advise you to use the packages from the Red Hat Rawhide directory on the Red Hat FTP server at **ftp.redhat.com**.

Don't let the recent lawsuits from SCO discourage you. There is a consensus within the Linux community that these lawsuits are without merit. Developers believe that even if there is any questionable code in the Linux kernel, they'll be able to substitute new code fairly quickly, which you could then easily download and install on various distributions, including Red Hat Linux.

This book is all you need to get started with Red Hat Linux on your network of Microsoft computers. In that spirit, Chapter 1, "Basic Linux Installation," and Chapter 2, "Installing Linux as a File Server," walk you through the installation process. Once you've completed these chapters, you'll have all the software you need to set up a Linux computer on a Microsoft network.

Chapter 3, "Setting Up Your File Server System," and Chapter 4, "Setting Up Your File Server's Users," show you how to set up Samba on your Linux computer. In Chapter 3, you'll examine the graphical tools included with Red Hat Linux that can help you configure Linux on a Microsoft network. In Chapter 4, you'll see how to make this work with users and groups on a network. In both chapters, you'll see how this works on the governing Linux configuration files.

In Chapter 5, "Connecting Linux Workstations," and Chapter 6, "Connecting Windows Workstations," you'll set up the workstations on your new network.

In the world of client-server networks, File and Print servers go together. In Chapter 7, "Configuring a Print Server," you'll see how to configure CUPS on Red Hat Linux as a print client and a print server on your new network.

Finally, Chapter 8, "Administration and Management," and Chapter 9, "System Backup," illustrate some survival skills that you'll use as a new Linux administrator. The information presented in these chapters isn't enough if you're serious about Linux, but it provides a start.

You can learn the skills you need to become a sophisticated Linux administrator from the other books in the *Linux Transfer* series, or from another book by this author, *Mastering Red Hat Linux 9*, from Sybex.

As you go through the book, you'll work as a Linux administrator. While Red Hat Linux GUI administrative tools are much improved, you'll be running a number of utilities from the command-line interface.

In addition, the Linux community is eager to help. If you identify yourself as a Microsoft Windows administrator, I've found that Linux users will bend over backwards to help you set up Red Hat Linux. Red Hat also maintains a number of active mailing lists where you can find help for your particular needs. See **www.redhat.com/mailman/listinfo**.

When you update Red Hat Linux with the latest available software, what you see may look somewhat different from what I illustrate in this book. As with Microsoft Windows Update, minor and even moderate variations for installed software are a fact of life.

Icons used in this book

Throughout this book, you'll see the following icons used to point out special notes, tips, and warnings.



Information of special interest, related topics, or important notes are indicated by this icon.



Tips—marked by this icon—include ideas for shortcuts, alternate ways of accomplishing tasks that can make your life easier or save time, or techniques that aren't immediately obvious.



This warning icon indicates pitfalls to watch out for, such as commands that have the ability to delete every file on your computer.

Chapter 1
Basic Linux Installation

In this book, I'll cover the tools you need to create a File and Print server on a Linux computer. The first chapter helps you figure out what it will take to move your file servers to Linux. It explains file servers and Linux distributions, and illustrates the basic steps you need to set up your computer for Linux. Many Linux distributions are available, but this book focuses on Red Hat Linux.

In this chapter I review the basic purposes of a file server, to help you define what you really need. I also take a look at Linux distributions. While Red Hat Linux is by far the most popular Linux distribution and the one used in this book, a number of other options are available. I follow up with guidance on hardware; just like Windows 2000/XP/2003, Linux may not work with all of your hardware.

Once you've identified the computer that you're going to use, it's best to plan ahead. You want to make sure Linux works with your hardware, and that you have sufficient space on your hard drives. In Linux, filesystems and swap space are organized in different hard-drive partitions. Once you've selected a partition configuration, you can prepare your hard drive. Finally, you can put this together with the basic steps you'll take to install Linux on your computer. The final steps are covered in Chapter 2, "Installing Linux as a File Server."

The same Linux distribution can be configured as a workstation or a server—you don't have to download (or buy) different versions for each. As a server, Red Hat Linux includes a number of tools that you can use to configure Web services, FTP, a firewall, and more. This book covers the tools you need to create a File and Print server on a Linux computer.

Basic file server functions

Before jumping into the wonderful world of Linux, it's a good idea to take a step back. What are you going to do with Linux? This book is targeted at people who want to use Linux as a File and Print server. Most of this book is dedicated to setting up a file server; I cover configuring a print server in Chapter 7, "Configuring Printers."

File servers are useful in a number of ways:

- **Central storage for individuals**: Many computer-savvy individuals have (or want) more than one computer, such as a desktop and a laptop. It's often most efficient to save your files on one computer, to make sure that you have one version of each file, and to ease your burdens when you back up your systems.
- **File sharing for workgroups**: Groups need a central file server. When users work together on a project, it's best if they work on the same version of a file. You should encourage the users in a workgroup to keep their individual files on a central server, so all you need to do for data security is to protect and back up the file server.
- **Organizations**: Organizations may need more than one file server. It's useful to think of a larger organization as a collection of workgroups, often organized into a Domain.

Selecting a distribution

Strictly speaking, Linux is just the kernel, the central module that serves as the interface between applications and computer hardware. A Linux *distribution* contains the other pieces of an operating system, which include several basic components:

- **Shell**: A shell includes the commands and utilities that allow you to navigate, command, administer, and configure Linux. The shell allows users to interface with the kernel. Linux distributions often come with a number of shells, and you can choose which one you want to use according to your personal preferences. The default shell used in this book is bash, the Bourne Again SHell.
- **Network services**: Linux is a clone of Unix, which was designed concurrently with the precursor to the Internet. Thus, Linux is built for networking in the Internet age and is well suited to modern workgroup and client/server networks.
- **Daemons**: Linux services run in the background and start on demand. Most of these services are known as *daemons* (pronounced "day-monz"). For example, Samba, the daemon that allows Linux to communicate on a network of Microsoft Windows computers, is smbd, in the /usr/sbin directory. It works hand-in-hand with the Linux NetBIOS name server daemon, nmbd. You can also activate the winbind daemon, winbindd, if you want to use a Windows NT/2000 server database of user names and passwords for the Linux computers on your network. (Daemons usually have a trailing 'd' in the file name.)
- **X Window**: The Linux graphical user interface (GUI) is built upon the program modules that are associated with the X Window. Strictly speaking, you don't even need a GUI to make Linux work like a Microsoft Windows Primary Domain Controller (PDC). Because most of you are more familiar with Microsoft Windows, I use GUI tools wherever possible in this book to configure Linux as a File and Print server.

Linux distributions also include components that you do not normally find in other operating systems, such as fully featured office suites, graphical programs, multimedia applications, and more.

Linux distributions are created by a number of different companies, organizations, and even volunteer groups. In this book I'll focus on Red Hat, which is the leading Linux distribution in the marketplace. But there are a substantial number of options in the world of Linux.



Strictly speaking, Linux is just the kernel, which allows the rest of the operating system to work with your hardware, allocate resources, and more. A number of the other components of the Linux operating system were developed by the Free Software Foundation, under the code name of GNU (GNU's Not Unix). Thus, many people in the Linux community refer to the GNU/Linux operating system.

Red Hat Linux

Red Hat Linux has released a number of different distributions, each aimed at a different band of customers, such as small users or large-scale enterprises. While the numbering systems are different and there are functional differences, most of the software in these Red Hat Linux distributions are, for the purposes of this book, identical.

Red Hat Linux 9

As of this writing, Red Hat Linux 9 is the latest Linux distribution from Red Hat. There are actually two versions of this operating system: Personal and Professional. You can download the software and source code for Red Hat Linux 9 Personal edition from **ftp.redhat.com** or one of the mirror sites listed at **www.redhat.com/download/mirror.html**. The installation software is also freely available as very large ISO files, suitable for copying to three writable CDs.

Alternatively, you can purchase the Personal or Professional editions, which come with a limited amount of support as well as a subscription to the Red Hat Network. The Professional edition includes more support as well as a multimedia applications CD.

Another advantage of Red Hat Linux is that even the Personal edition comes with various kernels customized for different CPUs, including two versions of the Symmetric Multiprocessing (SMP) kernel. In other words, you can install Red Hat Linux 9 on computers with more than one CPU.

Red Hat Enterprise Linux

Red Hat also includes several different distributions focused on the enterprise. These are designed for performance on larger (multi-CPU) computers on mission-critical systems. There are three basic versions of Red Hat Enterprise Linux:

- **Red Hat Enterprise Linux WS**: This operating system is designed for workstations on networks with the other Red Hat Enterprise distributions and is designed for desktop computers with up to two 32-bit or 64-bit CPUs.
- **Red Hat Enterprise Linux ES**: This entry-level server operating system is suited for larger networks, and is optimized for computers with up to two 32-bit CPUs but is limited to 4GB of RAM.
- **Red Hat Enterprise Linux AS**: This Red Hat advanced server operating system is optimized for larger servers with up to eight 32-bit or 64-bit CPUs and 16GB of RAM.

Each of these offerings includes various levels of support from Red Hat.

United Linux

United Linux is a consortium of four different Linux companies attempting to create a viable competitor to Red Hat Linux (**www.unitedlinux.com**). They include SuSE, SCO, Turbolinux, and Connectiva. So far, their combined efforts are focused on the enterprise with SuSE Linux Enterprise Server 8.1. While Red Hat is currently the market leader, keep an eye on United Linux. One of its member companies, SuSE, has a leading market share in Europe and is backed by IBM. Based on the recent lawsuits by SCO, I would be surprised if they are still a part of the United Linux consortium by the time this book goes to print.

Other Linux distributions

Red Hat Linux and United Linux are by no means the only available Linux distributions. As of this writing, nearly 200 different Linux distributions are listed at **www.linux.org**. I describe the benefits of a few of these distributions here.

- **Debian**: Debian Linux (**www.debian.org**) is a distribution created and supported entirely by volunteers. It is popular among the most dedicated Linux users. Debian includes its own package system for software installation, which is an alternative to the Red Hat Package Manager (RPM).
- **Lindows**: Lindows (**www.lindows.com**) is one of the Linux distributions focused on the desktop; you can get it pre-installed on computers sold by Wal-Mart. Unfortunately, it is not freely available for download like most other Linux distributions.
- **Mandrake**: Mandrake Linux (**www.linux-mandrake.com**) is loosely related to Red Hat, a practice which is allowed under the Linux General Public License (GPL). Mandrake includes a number of additional utilities that encourage ease of use.
- **SuSE**: While Red Hat has the dominant market share in Linux, SuSE (**www.suse.com**) is popular in Europe and is known as a more user-friendly distribution. SuSE also uses the RPM system. I particularly like SuSE as a user-friendly alternative on the desktop.
- **Turbolinux**: Similar to SuSE, Turbolinux (**www.turbolinux.com**) is more popular in East Asia, especially Japan.
- **Xandros**: Based on the Debian and the former Corel Linux distributions, Xandros (**www.xandros.com**) is another Linux distribution focused on the desktop. I find it to be quite user friendly; you can install Xandros in three basic steps. This distribution uses Debian's alternative package system. Unfortunately, Xandros is not freely available for download like most other Linux distributions.

A brief guide to installing Linux

This section provides a roadmap to the rest of this chapter, which is a brief guide to preparing your computer for Linux. I'll explain the basic first steps of the installation process here, but I won't cover actual software installation until Chapter 2, "Installing Linux as a File Server." When you install Linux, you should follow these basic steps:

1. Check your hardware. While I've never had any problems with basic components, not all hardware works with Linux. (For that matter, not all hardware works with Microsoft Windows XP/2003, either.)
2. Plan your partitions, the areas of your hard drives where Linux will be installed. I provide some typical scenarios in this chapter.
3. Make sure your partitions work within the Filesystem Hierarchy Standard (FHS).
4. Select an appropriate format for each partition. Later in this chapter I'll explain how this works with Disk Druid, the Red Hat installation partition-management tool.

5. If you're planning to dual-boot with Microsoft Windows, split your existing partitions as needed.
6. Learn the basic steps associated with installing Red Hat Linux.
7. Continue the installation using Chapter 2 as a guide.

1. Checking hardware

Linux does well with computer hardware. When installing Red Hat Linux, all you need to do in most cases is insert the first installation CD into the drive, set your computer to boot from the CD, and follow the prompts. Most hardware is detected and configured automatically during the installation process.

However, Linux does not work with all computer hardware. (For that matter, the latest Microsoft operating systems do not work with all computer hardware.) Thus, if you're planning to install Linux on a group of computers, it's worth the trouble to check your hardware against the Linux Hardware Compatibility Lists (HCL).

Red Hat Linux includes a guide to hardware compatibility at **hardware.redhat.com/hcl**. This page includes a link to a large database of hardware known to work with Red Hat Linux. It also includes links to additional lists of hardware compatible with most Linux distributions. Many of these components are not on the Red Hat HCL, but work well with Red Hat Linux. The information from many of these lists is summarized in the Linux Hardware Compatibility HOWTO document, available at **www.tldp.org/HOWTO/Hardware-HOWTO**.

Minimum requirements

For Red Hat Linux 9, your PC needs to meet at least the following requirements:

- **Pentium class CPU**: 200 MHz or higher for text mode; 400 MHz Pentium II or higher if you want to use the Linux GUI. If you're configuring a server for multiple users, the requirements increase accordingly.
- **Memory (RAM)**: At least 64MB if all you need is Linux in text mode on a workstation. You should have at least 128MB for a graphical workstation; 192MB is recommended. If you're configuring a server for multiple users, the requirements increase accordingly.
- **Hard disk space**: At least 475MB of disk space. If you install all the packages on the three Red Hat Linux 9 CDs, you'll need more than 5GB of disk space. You also may need additional hard disk space for a swap partition (typically twice the size of your RAM), third-party applications, and files for your users.
- **A "supported" video card and monitor**: You can install Red Hat Linux on many computers with many "unsupported" video cards and monitors if they can support Super VGA mode, also known as VESA mode. You may also need to know the maximum resolution and the horizontal and vertical refresh rates.
- **A keyboard and a mouse**: Red Hat Linux can detect many USB keyboards and pointing devices such as a mouse.

Remember, these are minimum requirements. For a File and Print server with a GUI desktop that can share files with Microsoft Windows computers, Red Hat Linux requires about 1GB of space for files. As you'll see, I've chosen a fairly minimal configuration that installs about 1.5GB of files. You need additional space for a swap partition, generally twice the amount of RAM on your computer. You'll also want space for third-party applications. You may also need space for user home directories, as well as directories shared by the users in a group. That does not include additional space required for a swap partition, files that you might share, users' home directories, or any additional applications that you might choose to install. This information is described in more detail in Chapter 2, "Installing Linux as a File Server."

Hardware checklist

It's not difficult to check your hardware against the available databases. Unfortunately, Red Hat Linux does not include a tool similar to the Windows 2000 Readiness Analyzer or the Windows XP Upgrade Adviser. Therefore, if you want to make sure your computer's hardware will work with Linux, you should set up your own database of hardware as a checklist:

- CPU type, speed
- RAM, in MB
- Hard drive size/available space
- CD/DVD type
- Keyboard style
- Pointing device (mouse), USB/Serial/PS2 connection, number of buttons
- SCSI adapter make/model
- Network card make/model/speed
- Telephone modem make/model/speed
- Video card make/model/memory
- Monitor make/model/allowable vertical and horizontal refresh rates

Your computer may include additional types of hardware. Information is available from a number of sources, including those listed in **Table 1**.

***Table 1**. Sources of Linux hardware information.*

Component	Web site	Description
Cameras	**gphoto.sourceforge.net**	Home page for developers of digital camera software for Linux.
FireWire (IEEE1394)	**www.linux1394.org**	Source for the latest IEEE1394 hardware information, also known as FireWire or iLink. This is experimental in the Linux kernel, but many IEEE1394 components work well.
Laptop computers	**www.linux-laptop.net**	Tips for installing and configuring Linux on a wide variety of laptop computers.

Component	Web site	Description
Mobile devices	**tuxmobil.org/howtos.html**	Information about using Linux on a wide variety of mobile devices, including laptops, handhelds, and more.
Modems	**www.linmodems.org**	Includes the latest support information for many Winmodems.
Network cards	**www.scyld.com/network**	Includes the latest information on support for network cards on Linux.
Printers	**www.linuxprinting.org**	Home page for the latest print drivers.
Scanners	**www.mostang.com/sane**	Source for scanner drivers and configuration help.
Sound cards	**www.alsa-project.org**	Home page of the Advanced Linux Sound Architecture project.
USB devices	**www.linux-usb.org**	Source for the latest information on Linux support of USB devices, including experimental support for USB 2.0 hardware.

Finding Linux drivers

If a Linux driver is available and ready for computer hardware, it's probably already incorporated into your Linux distribution. However, this may not be possible for the newest hardware.

Some manufacturers recognize the importance of Linux and include drivers with the disks that come with new hardware, or they make them available at the support pages on their Web sites. A few have even included drivers developed by the Linux community.

Don't despair if the maker of one of your components does not support Linux. A substantial number of drivers are ready for download from various Linux developers, tested by others, from some of the Web sites listed in Table 1. If all else fails, navigate to **www.google.com** and search using the make or model of your hardware and the word "linux." You may be pleasantly surprised!

Special issues

Some hardware is just a lot of trouble for Linux. The best examples are the so-called Winmodems and Winprinters, which incorporate Microsoft Windows driver libraries into their interfaces. Because Linux developers don't have access to the source code for proprietary Microsoft Windows drivers, they have had to "reverse-engineer" special drivers for a number of these components.

A lot of progress has been made with Winmodems. Red Hat Linux detects most of the Winmodems that I've used.



If you've used your hardware in Microsoft Windows, record its hardware settings in the Microsoft operating system. If Linux has trouble detecting your hardware, the IRQ, I/O, DMA, and COM port information can help. In Linux, COM1 corresponds to /dev/ttyS0, COM2 corresponds to /dev/ttyS1, and so on.

As of this writing, hardware that conforms to the USB 2.0 and IEEE1394 standards is supported on an experimental basis. But if you want Linux to work with hardware developed to

one of these standards, don't despair; many Linux developers have posted drivers and methods to make this hardware work online. And many of these solutions are fairly easy to implement.

Once you've evaluated your hardware, you're ready to plan the hard-disk partitions where you're going to install Linux.

2. Planning partitions

In Microsoft Windows, partitions seem easy. All you need is a single partition for all files, including a swap file. In contrast, most Linux installations use multiple partitions, which results in a more efficient use of space. One key to this is the Linux swap partition.

While Microsoft Windows associates drive letters with partitions, Linux associates (or "mounts," in Linux-speak) specific directories on different partitions. Each Linux directory includes specific types of files, based on the Filesystem Hierarchy Standard (FHS). Not all Linux directories are suitable for mounting; I explain this in detail later in the "Understanding the Filesystem Hierarchy Standard" section.

But before I can explain the FHS, I need to explain the standards for naming Linux partition devices.

Linux partition names

Before you can set up a Linux partition, you need to know something about the way partitions are organized and named. There are four alphanumeric characters associated with each hard disk partition. If the partition is on an IDE hard disk, the first two letters are *hd*. If it's on a SCSI hard disk, those letters are *sd*.

The third letter depends on the location of the hard disk. If it's an IDE hard disk, it depends on the position relative to the ATAPI controllers. For example, if it's the master on the primary controller, the third letter is *a*; if it's the slave on that controller, it's *b*; and so on. Similar criteria govern SCSI drives. If it's a CD or DVD player on an IDE or SCSI interface, that's it—because there are no partitions on CD or DVD players.

The fourth character is a number that depends on the type of partition. You can configure three different types of partitions on a hard disk:

- **Primary partition**: You can configure up to four different primary partitions on a hard drive. The partition with your boot files should be located on the active primary partition. Thus, if you have a primary partition, the fourth character in a Linux partition name is 1, 2, 3, or 4. The primary partitions on the first IDE hard drive can be hda1, hda2, hda3, or hda4.
- **Extended partition**: If you need more than four partitions on a hard drive, you'll want an extended partition. While you can't store data directly on this type of partition, you can further subdivide it into logical partitions. If you use the normal Red Hat installation program, an extended partition always has a fourth character of 4. For example, the extended partition on the second SCSI hard drive is labeled sdb4.
- **Logical partition**: You can subdivide the space in an extended partition into multiple logical partitions. The logical partitions that you create have a fourth character of 5 and above.

To make sure you understand this system, read **Table 2**. It includes several different examples of partition device names.

***Table 2**. Sample partition device names.*

Device	Partition
/dev/hda2	The second primary partition on the master IDE hard disk attached to the primary controller.
/dev/sdb	The second SCSI drive; may be the physical hard drive or a CD drive.
/dev/hdc	The master drive on the secondary IDE controller; may be a physical hard drive or a CD drive.
/dev/sdc5	The first logical partition on the third SCSI hard drive.

A swap partition

Linux normally uses a different virtual memory structure from Microsoft Windows. While Windows operating systems store less-frequently-used data in a swap file on one or more partitions, Linux normally uses a dedicated swap partition.

Swap partitions can supplement the RAM on your system to some extent. But remember, because a swap partition is on a hard drive, access is much slower than if it were stored in RAM. If you observe slow performance and can hear your hard drive working very hard, consider installing more RAM on your computer.

The ideal size of a swap partition is highly debatable. Generally, you'll want a swap partition that's twice the size of your RAM. If you have a small amount of RAM, such as 128MB, you might consider a swap partition up to four times the size of your RAM. Conversely, if you have a large amount of RAM, such as 1GB, a swap partition of the same size should be sufficient.

But as they say, your mileage may vary. The swap partition that works best for you depends on your hardware and the demands of other computers on your system.

RAID and LVM

When creating partitions, two other things to consider are security and flexibility. The right Redundant Array of Independent or Inexpensive Disks (RAID) can help make sure that your computer keeps working even if a single hard disk fails. RAID is covered in more detail in Chapter 9, "System Backup."

Logical Volume Management (LVM) allows you to set up partitions in chunks. As your needs evolve, it allows you to change the effective size of your partitions. In that way, it is functionally similar to the third-party tools System Commander and Partition Magic, which can also resize Linux partitions that are formatted to the ext2 or ext3 filesystems.

Later in this chapter, I briefly review the process of creating RAID and LVM arrays.

3. Understanding the Filesystem Hierarchy Standard

Before you can create a good plan for your partitions, you need to know how files and directories are organized. Then you can select appropriate Linux directories to mount on a partition, a RAID array, or an LVM volume.

The Filesystem Hierarchy Standard (FHS) is the basis for organizing directories and files in Unix-style operating systems, including Linux. For more information, see **www.pathname.com/fhs**. Not all of the directories in Red Hat Linux conform to the FHS. While the information in **Table 3** may seem long, read through it. It can help you avoid mistakes that can stop your Linux computer.



There are several directories that you should not mount separately; for example, if you mounted the /bin directory on a separate partition, you might not be able to use the commands in that directory to rescue your data from a corrupt system.

***Table 3**. Basic Red Hat Linux directories.*

Directory	Function
/	The top-level directory is known as "root." All other directories are subdirectories. Any directory not mounted on a separate partition is included in the root (/) directory's partition.
/bin	Includes the basic utilities that you can run at the command line, such as **ls**, **cp**, and **mv**. Never mount this directory on a separate partition.
/boot	Contains critical files and commands for the Linux boot process, including the Linux kernel and a boot loader. You should normally mount this directory on a separate partition.
/dev	Adds the device files. For example, /dev/hda1 is the device associated with the first primary partition on the first IDE hard drive. Drivers for installed devices are stored in the /lib directory. Never mount this directory on a separate partition.
/etc	The location for most Linux configuration files. For example, /etc/passwd contains basic user information, and /etc/samba/smb.conf contains basic Samba server configuration data. Do not mount this directory separately.
/home	Includes the home directories of all users except the administrative user, root. If you want to make sure that your users' files don't overflow into areas needed by Linux, consider mounting this directory on a separate partition.
/initrd	Adds an empty directory used by Red Hat Linux for initial RAM disk files. Do not add anything to this directory. Do not mount it on a separate partition. Do not delete it!
/lib	Contains program libraries and hardware drivers. Do not mount this directory on a separate partition.
/lost+found	The location for orphan files, such as those recovered from a disk check with the **fsck** command.
/misc	Adds a common mount point for directories shared from remote servers.
/mnt	Normally used as the mount point for removable media. For example, /mnt/cdrom is the usual location for files from a CD.
/opt	Includes the standard location for a number of third-party applications. Consider mounting this directory separately.
/proc	Adds a virtual directory of kernel-related processes and detected hardware. For example, /proc/dma includes direct memory address assignments. Because this takes up no room on a hard drive, you need not mount this directory separately.
/root	Includes the home directory of the root user. Home directories for all other users are contained in /home. Never mount this directory separately.
/sbin	Contains many commands used for system administration. Never mount this directory separately.
/tmp	Contains the dedicated storage location for temporary files. By default, Red Hat Linux deletes any files in this directory that have not been accessed for 10 days.
/usr	Adds small programs and utilities normally accessible to all users.
/var	Includes variable data such as print spools, log files, and FTP uploads. If you anticipate any of these files to be very large, it's a good idea to mount this directory on a separate partition.



Remember, the home directory of the root user is /root, which is a subdirectory of the top-level Linux directory, /, which is known as the root directory.

Based on Table 1, you may wish to mount the following directories on separate partitions: /boot, /home, /opt, /tmp, and /var. And don't forget the swap partition. Once you've selected the directories that you'll mount on different partitions, you'll need to format each partition to a specific filesystem. Red Hat Linux formats partitions to your selected filesystem during the installation process.

4. Selecting a filesystem format

Just as Microsoft Windows partitions may be formatted to the FAT16, FAT32, or NTFS systems, Linux includes its own filesystem formats. One primary advantage is efficiency; unlike Microsoft filesystems, Linux filesystems can work well even when very little space is available on a partition. For example, I've run Linux partitions that were nearly 99 percent full with no visible loss of speed.

You can use a number of different formats on a Linux system; I cover four of them here. Two of them, ReiserFS and XFS, were introduced with Linux kernel version 2.4 and are therefore fairly new.

Linux extended filesystems

Two major Linux extended filesystems are most commonly used today: the second and third extended filesystems, which are known as ext2 and ext3, respectively. While the current default for Red Hat Linux is ext3, it is fully compatible with the default for older versions of Linux, ext2.

The only difference is that ext3 includes a journal, which speeds recovery from a problem such as a sudden power failure. In this way, ext3 is functionally similar to Microsoft's NTFS5, first introduced for Windows 2000. It's easy to convert partitions between ext2 and ext3. Therefore, ext3 is appropriate for later versions of Red Hat Linux because it allows you to use legacy data on ext2 partitions.

One major problem with the ext2 and ext3 formats is that file sizes are limited to 2GB. This can be a problem for very large databases or even the log files associated with the largest online merchants.

ReiserFS

The ReiserFS filesystem includes fast journaling. In other words, the time it takes to recover from a problem such as a sudden power failure is less than you might experience for an ext3-formatted partition. ReiserFS is based on a concept known as "balanced trees," which enhances performance if most of your files are very small and very large. It is also more space efficient, because it can store more than one file per disk block. It's sponsored by the U.S. Defense Advanced Research Projects Agency, which is a testament to its reliability. For more information on ReiserFS, see **www.namesys.com**.

One additional advantage is that you can increase the size of a ReiserFS filesystem dynamically, up to the current size of the partition.

XFS

The XFS filesystem was developed by Silicon Graphics (SGI) as an early journaling filesystem for SGI's IRIX operating system. Like Linux, IRIX is a clone of Unix. Unlike Linux, IRIX is primarily used for computers with 64-bit processors. SGI has ported or adapted XFS for Linux.

XFS has a number of advantages over ext3. The maximum file size is currently 9×10^{18} bytes, which is about 4,500,000 times the size of the largest allowable ext3 file. If you find this to be too small, SGI plans to eliminate this limit in future releases of XFS. This filesystem is also well suited to high-bandwidth transfers.



Don't confuse the XFS filesystem with the Linux X Font Server; both use the same lowercase acronym: xfs.

Working with other filesystems

Once you've connected Linux and Microsoft Windows computers in a network, file transfers between the two systems are nearly seamless. But you need to watch out for several differences.

- Linux file names are case sensitive. For example, Linux sees Chapter1.doc as a different file from chapter1.doc.
- Linux file names don't require specific extensions. For example, whether Linux executes a program such as "program.exe" or "program" does not depend on the EXE extension, but on the permissions associated with the file.
- If you have Linux and Microsoft Windows on the same computer, Linux can see the files in the Windows directories. However, Microsoft Windows can't see the files in the Linux partitions.

> **!** *If you install Linux and Microsoft Windows on the same computer, don't write any files to an NTFS partition from Linux. You could easily corrupt the data on the NTFS partition. This does not apply if you're writing to an NTFS partition on a remote computer.*

For your reference, these filesystems are summarized in **Table 4**.

***Table 4**. Partitions for installing Linux on a small hard drive.*

Filesystem format	Description
ext2	The second extended filesystem is an older Linux filesystem, in common use through Red Hat Linux 7.2.
ext3	The third extended filesystem is now the default Red Hat Linux filesystem; it's essentially the same as ext2 but has a journal for quick recovery from sudden failures.
reiserfs	The Reiser filesystem supports fast journaling, is faster because of its "balanced trees," and makes more efficient use of space.
swap	Linux swap partition. A dedicated area on the hard drive for virtual memory.
vfat	A common Linux description of a Microsoft Windows FAT16 or FAT32 filesystem. (The difference is immaterial to Linux.)

Filesystem format	Description
xfs	The xfs filesystem was developed for 64-bit operating systems and has no practical limit on file size.
LVM	A logical volume management filesystem can be organized into chunks that can be reallocated between different partitions.
RAID	Red Hat Linux supports software RAID, which applies the basic rules of hardware RAID to individual partitions.

Typical partition scenarios

This section illustrates four different scenarios that you might use in configuring a Linux file server: a computer with a small hard drive; one with a decent-sized hard drive where you want to limit the space available to users in their home directories; a computer with a decent-sized hard drive where you've also installed Microsoft Windows; and finally a computer with a hard drive that's divided into a substantial number of partitions.

The scenarios in this section are functionally equivalent to the way that some Windows administrators divide hard disks into C:\, D:\, E:\, and additional drives for certain categories of software.

In addition, Red Hat Linux normally formats the partitions with mounted directories to the ext3 filesystem. It formats swap partitions to the Linux swap filesystem.

Scenario 1

In the first scenario, you're just experimenting with Linux. You have an older computer with a 2GB hard drive and 128MB of RAM. That's certainly not enough room to install the 5GB of packages available to Red Hat Linux. But it is still a good idea to configure separate partitions for the /boot directory and swap space. In this scenario, I'd allocate the space on the hard drive as shown in **Table 5**.

***Table 5**. Partitions for installing Linux on a small hard drive.*

Allocation	Size
/boot	100MB
swap partition	256MB
/	1.65GB

Scenario 2

In the second scenario, you're installing Linux on a production computer. It has a 200GB hard drive and 1GB of RAM. That's certainly more than enough space for all the files that you can install from Red Hat Linux. However, you want to make sure that your users are reasonably disciplined about their file storage habits. Assume that you don't expect more than 100 users and will use quotas (described in Chapter 4, "Setting Up Your File Server's Users") to limit their usage to 1GB each. In addition to the partitions described in the previous scenario, you'll need a separate partition for the /home directory. In this scenario, I'd allocate the space on the hard drive as shown in **Table 6**.

***Table 6**. Partitions for installing Linux as a file server with a large hard drive.*

Allocation	Size
/boot	100MB
swap space	1GB
/	99GB
/home	100GB

While kernels can be large, it's rarely necessary to allocate more than 100MB to the /boot directory. If you're also using this computer for other purposes, you might consider creating additional partitions, such as /var, to limit the effect of FTP uploads or Web server log files.

Scenario 3

In the next scenario, you don't have enough computers but want to test Linux. You want to install Linux on the same computer that contains Microsoft Windows. This is sometimes known as a *dual-boot* configuration. You have a 30GB hard drive and 256MB of RAM. After defragmenting your hard drive, you decide that all you need for Microsoft Windows is 15GB of space.



Because of fragmentation, I try to keep at least one-third of the hard drive free on partitions where I've installed Microsoft Windows.

Because you'll have two operating systems on this computer, I'm assuming that it won't be used as a mission-critical production computer. While you still have some leeway for additional partitions, I'm also assuming a configuration similar to scenario 1. I'd allocate the partitions as shown in **Table 7**. Later in this chapter, I'll show you how to use the First Interactive Partition Splitter (FIPS) to split a hard drive.

***Table 7**. Partitions for installing Linux in a dual-boot configuration.*

Allocation	Size
Microsoft Windows	15GB
/boot	100MB
swap space	512MB
/	14.5GB

Scenario 4

In the final scenario, you're installing Linux on a production computer. You've been told to isolate a number of directories in their own partitions, so excesses in files don't affect other directories or the performance of the system. You have a 400GB hard drive and 1024MB of RAM.

You want to isolate separate partitions for the /home directory to limit the space taken by your users. You want a separate partition for /opt, dedicated to third-party applications. If you have a Web server and an online business, a separate partition for /var prevents log files from

overwhelming your system. Finally, downloads are commonly stored in /tmp; a separate partition for this directory ensures that users don't overwhelm your system with downloads.

Naturally, you'll still want separate /boot and swap partitions. In this scenario, I'd allocate the space on the hard drive as shown in **Table 8**.

***Table 8**. Partitions for installing Linux on a production computer.*

Allocation	Size
/boot	100 MB (that's megabytes)
/home	100 GB
/opt	50 GB
/var	100 GB
/tmp	100 GB
/	50 GB
swap space	2000MB

5. Preparing partitions and hard drives for dual booting

Before you can install Linux on your computer, you need free space on your hard drive. To get the full benefit of Linux, you need to set up separate partitions in that free space. This section assumes that you want to set up Linux in a dual-boot configuration with Microsoft Windows. If you have a dedicated computer or empty hard drive ready for Linux, feel free to jump ahead to the section "Red Hat's Disk Druid."

The dual-boot option

There are a number of ways to set up a dual-boot between Linux and Microsoft Windows. You can use third-party proprietary tools such as System Commander or Partition Magic. If you're using one of the Microsoft 32-bit operating systems (Windows NT/2000/XP/2003), you can even use Microsoft's NTLDR. If you're installing Linux, the standard way is to use the boot loader. The default Red Hat Linux boot loader is known as GRUB and is shown in **Figure 1**. When you see this screen, you can use the up and down arrow keys to select the desired operating system from the menu.

If you've installed Linux in an empty partition, distributions such as Red Hat Linux will automatically set up the dual-boot. Once Linux is installed, your computer reboots, and the next thing you see is a menu similar to that shown in Figure 1. If you want more information on how to configure a boot loader, see my book from Sybex, *Mastering Red Hat Linux 9*.

Splitting an existing partition

Usually, computers with Microsoft Windows installed don't have enough usable free space. In many cases, the entire hard drive is allocated as one partition, which you probably know as the C: drive. Even with the piles of office suites and graphical applications available with Linux, it does not require a great deal of room when compared to the latest hard drives.

While you can purchase a third-party tool to split a partition, it usually isn't necessary. Most Linux distributions, including Red Hat Linux, include the First Interactive Partition Splitter in the form of FIPS.EXE. It works fine with *Primary* partitions formatted to the Microsoft FAT16 or FAT32 filesystems. However, FIPS.EXE cannot split an NTFS partition.

Like other open-source products, it comes with absolutely no warranty, so use it at your own risk! That said, I've used it numerous times without problems.



***Figure 1**. GRUB, the default Red Hat Linux boot loader, allows you to dual-boot between Linux and Microsoft Windows.*

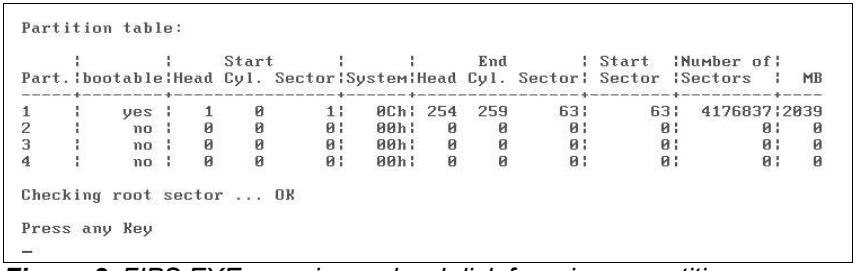
The following procedure assumes that you haven't used up all four primary partitions and don't have an extended partition on the subject hard drive. It also assumes that you'll be splitting the first primary partition, which in Microsoft Windows is allocated to the C: drive.

1. Run the **FIPS.EXE** command from a DOS prompt. If you need to abort the utility, press **Ctrl-C** at any time.
2. Prepare your hard drive in Microsoft Windows. Use the Microsoft disk defragmenter program on your Windows operating system. This maximizes the available space at the end of the partition.
3. Prepare a DOS boot disk. My favorite for this purpose is associated with Windows 98. You can get a copy of this boot disk from **www.bootdisk.com**. Copy FIPS.EXE and RESTORRB.EXE to that disk. In Red Hat Linux, these files are on the first installation CD, in the /dosutils directory.



You may also want to print out the FIPS.DOC file, which is in the same directory as FIPS.EXE. Instructions for using RESTORRB.EXE are contained there.

4. Reboot your computer with the DOS boot disk in the floppy drive.
5. When your computer boots to the A: drive, run the **FIPS.EXE** command. Press a key when prompted. If FIPS.EXE detects two hard drives on your computer, you'll be asked to select between the drives. (You can't boot Linux from a third physical hard drive.)
6. You should see a table listing the partitions on your hard drive, similar to **Figure 2**. As you can see, there is only one primary partition configured on this particular drive. If there's more than one primary partition, you'll be prompted to choose. If all four partition slots are configured, FIPS.EXE aborts, because it splits primary partitions.



```
Partition table:

     ¦        ¦     Start      ¦      ¦      End       ¦ Start  ¦Number of¦
Part.¦bootable¦Head Cyl. Sector¦System¦Head Cyl. Sector¦ Sector ¦Sectors  ¦  MB
-----+--------+-----------------+------+-----------------+--------+---------+----
1    ¦     yes ¦   1    0      1¦  0Ch¦ 254  259     63¦      63¦ 4176837¦2039
2    ¦      no ¦   0    0      0¦  00h¦   0    0      0¦       0¦       0¦   0
3    ¦      no ¦   0    0      0¦  00h¦   0    0      0¦       0¦       0¦   0
4    ¦      no ¦   0    0      0¦  00h¦   0    0      0¦       0¦       0¦   0

Checking root sector ... OK

Press any Key
_
```

***Figure 2**. FIPS.EXE examines a hard disk for primary partitions.*

7. FIPS.EXE then scans the selected partition. After listing basic information, it offers to make backup copies of the root and boot sectors of your hard disk. Accept the offer! Let FIPS.EXE store the backup on your boot floppy. Make a note of the file name that is written, probably a:\rootboot.000. It's the best insurance against mistakes.
8. Based on the empty space you have available, you can now split the selected partition. Use the arrow keys to allocate space between old and new partitions. You'll see a screen similar to the following:

```
Old partition          Cylinder          New partition
4150.6 MB              290               6120.3 MBdf
```

9. If your old partition contains Microsoft Windows, remember to leave enough room for the swap file and fragmentation. When you're satisfied with the result, press Enter to confirm the two new partitions.
10. FIPS.EXE tests the new partition. If everything works, you'll see a revised partition table similar to Figure 2. You can continue or re-edit the partition table with the **c** or **r** commands. If you're satisfied with your changes, press **c** to continue.
11. Write your new partition scheme to disk. If all went as planned, you'll be able to identify this empty partition during the Linux installation process. Alternatively, you can delete and reconfigure the space associated with the new partition by using the Linux version of fdisk.

Basic filesystem terms

Look over the following terms. They can help you understand what happens with partitions, RAID, and LVM.

Volume: An area of disk space to which you can assign a Microsoft drive letter such as C: or a Linux directory such as /boot. A volume is usually equivalent to a partition, unless you've configured two or more partitions to work together in a RAID 0 array.

Mount point: The directory, such as /boot, where you're mounting a partition, RAID device, or logical volume.

RAID: Redundant Array of Independent (or Inexpensive) Disks. With hardware RAID, there are two or more physical hard drives working together. Hardware RAID requires third-party software not included with Linux.

Software RAID: RAID that uses partitions instead of physical hard drives. To be effective, each partition in a software RAID array should be on a different physical hard drive.

RAID device: An array on which you can mount a partition. RAID arrays can be configured in several standard ways: Three are known as RAID 0, RAID 1, and RAID 5. In Red Hat Linux, RAID devices are named /dev/md0, /dev/md1, and so on. RAID devices require two or more hard drives or partitions.

RAID 0: Fast, but not good for data security. RAID 0 allows you to combine the space on two or more partitions or hard drives in a single volume. Also known as "disk striping without parity."

RAID 1: Slow, but excellent for data security. RAID 1 copies the same data on two independent partitions or hard drives in a single volume. Also known as "disk mirroring."

RAID 5: Excellent for data security; moderate speed. Requires three or more partitions or hard drives. RAID 5 includes additional data in parity bits, which can help you rebuild data from any corrupt partition or hard drive.

Physical Volume (PV): A primary or logical partition that you've assigned for Logical Volume Management (LVM).

Physical Extent (PE): A physical volume that is subdivided into equal blocks.

Logical Extent (LE): Every LE corresponds to a PE. Physical Extents are then organized into logical volumes.

Logical Volume (LV): A bunch of LEs grouped together, which can then be used to mount a directory—just like a standard partition.

Volume Group (VG): All of the LVs on your system for a volume group.



RAID 0, RAID 1, and RAID 5 are the only versions of RAID supported by Red Hat Linux.

Adding and deleting partitions with fdisk

The standard tool for adding and deleting partitions in Linux is fdisk. The Microsoft version of this utility, FDISK.EXE, can't be used to create more than one primary partition. Because Red Hat includes the Disk Druid tool, it is no longer necessary to use fdisk to prepare partitions for installation. However, Disk Druid is not available once Red Hat Linux is installed, so you still need to know how to use fdisk when you add a new hard drive.

The Linux fdisk utility can manage more than 100 different types of partitions, including various Microsoft, Novell, and of course, Linux types. Before I show you an example of how to

create a new partition, review the five basic Linux partition types listed in **Table 9**. The number associated with each type is the designation within fdisk.

***Table 9**. Linux partition types.*

Type	Hex code	Purpose
Linux	83	Suitable for standard Linux data formats, including ext2, ext3, ReiserFS, and xfs.
Linux Extended	85	Associated with an extended partition that contains Linux logical partitions.
Linux LVM	8e	Used for partitions where you're allocating the space to LVM volumes.
Linux RAID	fd	Assigned to partitions in a software Linux RAID array.
Linux swap	82	For the partition dedicated to Linux swap space.

Now let's look at configuring partitions on a new hard drive. For example, if you've just installed a new SCSI hard drive and run the **fdisk -l** command, you should see the partition tables as configured on your hard drives, similar to **Figure 3**.

```
[root@NoPaws root]# fdisk -l

Disk /dev/sda: 4293 MB, 4293596160 bytes
255 heads, 63 sectors/track, 522 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sda1   *         1        13    104391   83  Linux
/dev/sda2            14       474   3702982+  83  Linux
/dev/sda3           475       522    385560   82  Linux swap

Disk /dev/sdb: 2146 MB, 2146798080 bytes
128 heads, 32 sectors/track, 1023 cylinders
Units = cylinders of 4096 * 512 = 2097152 bytes

Disk /dev/sdb doesn't contain a valid partition table
[root@NoPaws root]#
```

***Figure 3.** Checking partitions.*



*Depending on the defaults for your user name, you may get a "command not found" error. In that case, use the full path to a command, such as /sbin/fdisk /dev/sdb. For example, if I want to find the full path to fdisk, I run the command **locate fdisk | more**.*

You can also see from Figure 3 what happens just after you've installed a new hard drive. Note the name; it's /dev/sdb. From the discussion earlier in this chapter, you should realize that it's the second SCSI hard drive on your computer. Now use the following command to start configuring that hard drive:

```
# fdisk /dev/sdb
```

It starts fdisk with the following prompt:

```
Command (m for help): n
```

Now follow these steps:

1. At the **fdisk** command prompt, run the **n** (for new partition) command. (Other important fdisk command options are shown back in Table 7.)

```
Command action
   e   extended
   p   primary partition (1-4)
```

2. Press **p** to start creating a primary partition on the new hard drive. When prompted, select the first available partition number, which should be **1** for a newly installed hard drive.
3. Assign the first cylinder for the desired primary partition. fdisk automatically starts the given range with the first available cylinder. Select the first cylinder of your choice. You should see a prompt similar to this:

```
Last cylinder or +size or +sizeM or +sizeK (1-1023, default 1023):
```

4. There are several ways to define the size of the first primary partition. Assuming your hard disk has sufficient space, just enter the desired size of the partition; for example, the **+5000M** command prompts fdisk to create a 5000MB partition.
5. Repeat steps 1–4 as desired to create additional partitions on the new hard drive. You can review the changes at the **fdisk** prompt by using the **p** command.
6. If you're satisfied with your changes, use the **w** command to write the partition table to the hard drive; use the **q** command to exit fdisk without changing the partition table.

Additional command options are listed in **Table 10**.

***Table 10**. Major command options at the **fdisk** prompt.*

Option	Function
a	Toggles the bootable flag. You need to set this flag on the partition where you mount the /boot directory.
d	Starts the process of deleting a partition. You're prompted for the partition number.
l	Lists partition types known to fdisk.
m	Lists available fdisk commands.
n	Starts the process of creating a new partition. You're prompted for the partition number and size.
p	Shows currently configured partitions.
q	Quits fdisk without saving changes.
t	Starts the process of changing the type of the partition.

By default, fdisk creates partitions that you can then format to a Linux native type such as ext2, ext3, ReiserFS, or xfs. It's easy to format the new partition. For example, if you want to format /dev/sdb1 to the default Red Hat Linux ext3 filesystem, just run the following command:

```
# mkfs.ext3 /dev/sdb1
```

But you may want to set up a different type of partition. Assume that you want to set up the first primary partition on the new hard drive as a swap partition. The commands shown in **Figure 4** reconfigure the first partition on the second SCSI hard drive as a swap partition.

```
[root@NoPaws root]# fdisk /dev/sdb

Command (m for help): p

Disk /dev/sdb: 2146 MB, 2146798080 bytes
128 heads, 32 sectors/track, 1023 cylinders
Units = cylinders of 4096 * 512 = 2097152 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sdb1             1       239    489456   83  Linux
/dev/sdb2           240       740   1026048   83  Linux

Command (m for help): t
Partition number (1-4): 1
Hex code (type L to list codes): 82
Changed system type of partition 1 to 82 (Linux swap)

Command (m for help): p

Disk /dev/sdb: 2146 MB, 2146798080 bytes
128 heads, 32 sectors/track, 1023 cylinders
Units = cylinders of 4096 * 512 = 2097152 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sdb1             1       239    489456   82  Linux swap
/dev/sdb2           240       740   1026048   83  Linux

Command (m for help):
```

***Figure 4**. Assigning a Linux swap partition.*

Red Hat's Disk Druid

If you're installing or upgrading Red Hat Linux, you have an alternative to fdisk, known as Disk Druid. It can help you create and format Linux ext3 and swap partitions automatically. When you install Red Hat Linux graphically, you'll see the Disk Druid screen as shown in **Figure 5**. (Text-mode installation leads to a similar screen, which does not support LVM configuration in Red Hat Linux 9.)

Using Disk Druid, you can add, edit, and delete the partitions that you need. You can also configure RAID arrays and LVM volumes. Four of the corresponding buttons open a dialog. There isn't enough room in this book to analyze each option step by step; the following sections list the information that you need to perform each function. I leave it up to you to fill in the proper information in each dialog.

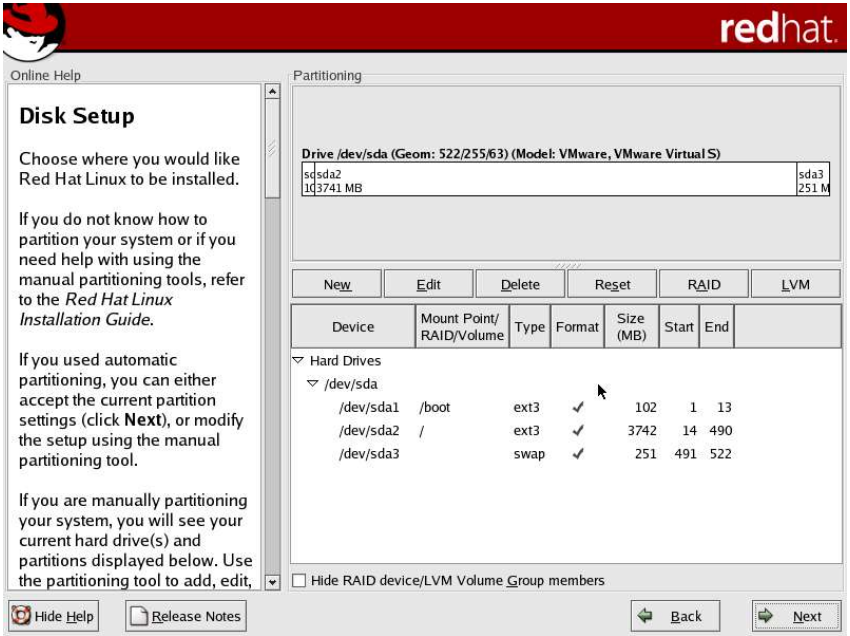


Figure 5. *Red Hat's Disk Druid.*

The Delete button is self-explanatory; if you want to delete a partition, highlight it and then click Delete. If you make a mistake, you can click Reset and restore the current configuration.

Adding or editing a partition

When you click New or Edit in the Disk Druid screen, you can add or edit a partition using any empty space that's available on your hard drives. To add or edit a partition, you'll need the following information:

Mount point: The directory such as /boot or /home to which you're mounting this partition. If you've selected a RAID, LVM, or swap filesystem type, a mount point is not required. Shortly, you'll amalgamate RAID and LVM partitions into devices with a mount point.

File system type: The way you want to format this partition. The default is ext3. You'll also need to format a partition to the Linux swap filesystem. Options include ext2, ext3, LVM, software RAID, swap, and vfat. These options are detailed in **Table 11**.

Allowable drives: If you're setting up Linux partitions on more than one physical hard drive, you can specify the hard drive where you want this particular partition.

Size: Naturally, you'll want to know how much space you need for the directory that's mounted on this partition.

Use of empty space: If you have empty space when you're done adding partitions to your hard drive, you can let it grow up to a specific maximum size, or fill all available space on the hard disk.

Primary partition: If you've mounted the /boot directory separately, it has to be on a primary partition.

Bad blocks: Before Red Hat Linux installs packages, it checks for bad blocks and then formats your partitions.

Software RAID

If you want to create a software RAID device, the first step is to create two or more software RAID partitions. They should be approximately equal in size. Each partition should ideally be on a different physical hard drive. Once you've used Disk Druid to create a sufficient number of RAID partitions, click RAID. This opens the RAID Options dialog. Select the "Create a RAID device" option and click OK.

This opens the Make RAID Device dialog shown in **Figure 6**.

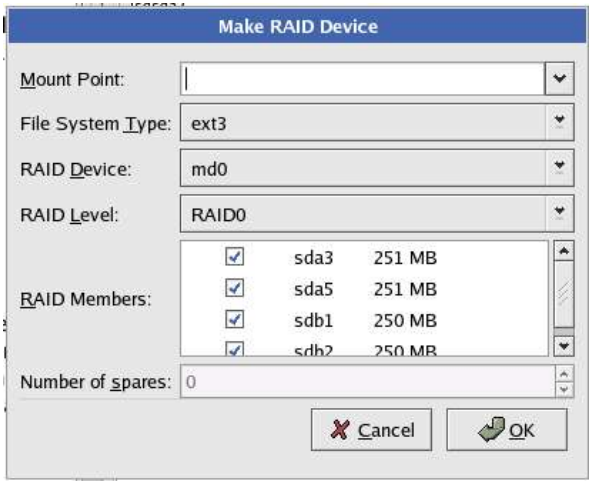


***Figure 6**. Configuring a software RAID device.*

Now you can set up a RAID 0, RAID 1, or RAID 5 device, and mount the directory of your choice on that system. The options shown in Figure 6 are summarized in Table 11.

***Table 11**. Options when configuring a software RAID device.*

Option	Purpose
Mount Point	The directory to be mounted on the RAID device.
File System Type	The format to be used on the RAID device. Options include ext2, ext3, LVM, swap, and vfat.
RAID Device	The file name associated with this RAID device; the options range from md0 to md15.
RAID Level	Red Hat Linux allows you to select from RAID 0, RAID 1, or RAID 5.
RAID Members	Lists all configured and free software RAID partitions.
Number of spares	RAID 1 requires a minimum of two partitions; RAID 5 requires a minimum of three partitions. Any spares will automatically replace any partitions that go bad.

LVM

When you click LVM, Red Hat Linux opens the Make LVM Volume Group dialog, which lists the configured LVM physical volumes. Click Add to create a logical volume where you can mount the directory of your choice, as shown in **Figure 7**. As discussed earlier, logical volumes are flexible; partitions configured into LVs can be easily resized.

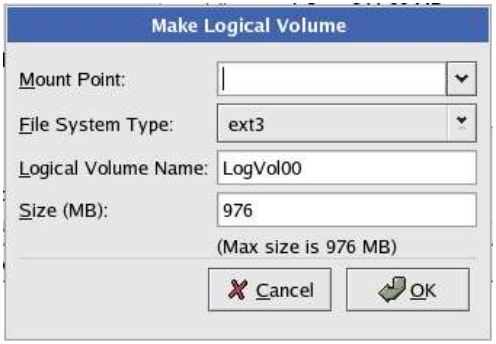


***Figure 7**. Creating a Logical Volume.*

As I mentioned before, the mount point is the directory to be mounted on the volume. The filesystem options are limited to ext2, ext3, and swap. You can set the Logical Volume Name of your choice. Any space that you don't assign to this LV is available for another LV.

6. Basic steps to installing Red Hat Linux

Installing Red Hat Linux is not difficult. Except for what I've just covered in configuring partitions, I think anyone could install Red Hat Linux on their computers with the guidance from this section and the next chapter. The steps required to install Red Hat Linux from a CD are slightly different from those you can use to install this operating system from a network source.

Local installation steps

When you start the Red Hat Linux installation process, the basic steps are straightforward. In most cases, the installation program automatically detects your hardware; all you need to do is confirm the selection. The following procedure is based on Red Hat Linux 9, but it hasn't changed greatly from previous versions of Red Hat Linux. My previews of future versions of Red Hat Linux suggest that minor changes are coming. So if you are not using Red Hat Linux 9, the following steps aren't exact.

Remember, these are just the first installation steps from a CD; I describe the guts of the installation process in more detail in Chapter 2, "Installing Linux as a File Server."

1. Set your computer's BIOS to boot from the CD.



If you can't boot from your CD, you can create a boot floppy. Run the RAWWRITEWIN.EXE utility in Microsoft Windows; it's on the first Red Hat installation CD, in the /dosutils/rawritewin directory.

2. Insert the first installation CD into the drive and restart your computer.
3. When you see the "boot:" prompt, press Enter. If your computer can't support at least SVGA-level graphics, type text and press Enter to run the Red Hat Linux installation program in text mode. The steps are nearly identical.
4. You'll see a series of screens that will help you configure your Linux installation. Help on many screens is available in the left-hand pane. When you finish the tasks on each screen, click Next.



In most cases, you can stick with the default option during the Red Hat Linux installation process. It detects most hardware automatically, and the defaults allow you to install Red Hat Linux adequately in most hardware configurations.

5. Select a language to use during the installation process. This is not necessarily the language that gets installed with the operating system.
6. Select from the available keyboards. If Red Hat detected your keyboard, it should already be highlighted.
7. Select the mouse that most closely matches your hardware. If you have a mouse wheel, press down on it. If it clicks, you should be able to use it as a third button. Otherwise, if you have a regular two-button mouse, select the "Emulate 3 buttons" option.
8. If you're trying to install Red Hat Linux on a computer that already has Linux installed, you might see an upgrade screen. (These steps assume that you're installing a fresh copy of Red Hat Linux.)
9. Select from the four available installation types: Personal Desktop, Workstation, Server, or Custom. Either a Server or a Custom installation is well suited for a Linux File and Print server. For the purposes of this procedure, select a Custom installation.
10. Allow Red Hat Linux to automatically partition your hard drives. You can change them later by using Disk Druid. Allow the installation program to remove just the Linux partitions on your computer. Select the "Review (and modify if needed) the partitions created" option.



If you selected a Server installation, Red Hat deletes all partitions by default, even if you have Microsoft Windows or another operating system installed on one of those partitions.

11. When you see the Disk Druid screen from Figure 5, inspect the choices made by Red Hat's automatic partitioning. Feel free to make changes based on the criteria described in this chapter.
12. Configure a boot loader, which is the first menu that you see when you boot your computer. This step is especially important if you want to access a second operating system such as Microsoft Windows on the same computer. Red Hat Linux allows you to configure the default operating system and labels. If you choose the default GRUB

boot loader, it's a good idea to protect it with a password; otherwise a *cracker* can break into your system and change your administrative password, which belongs to the root user.



In the world of Linux, crackers are the people who intend to do harm to your system. Don't confuse crackers with Linux hackers, good people who tinker to improve your software.

13. Configure any detected network cards. You can configure your cards to get IP address information from a DHCP server on a local or remote network (via BOOTP), or you can set IP addresses manually.
14. Configure a firewall for your computer, which is described in more detail in Chapter 2, "Installing Linux as a File Server." Configure the language support associated with this computer. You can switch between default languages in Red Hat Linux after installation by using the redhat-config-languages utility.
15. Set the time zone associated with your computer. In Chapter 2, I'll show you how to synchronize your computer with a central time server.

Other steps, including configuring password servers (authentication), firewalls, software selection, and graphics configuration, are covered in Chapter 2.

Network installation steps

You can also install Red Hat Linux over a network. If your Internet connection is fast enough, you could hypothetically install it directly from the Red Hat FTP site or one of the mirror sites. All you need is the directory path to the /RedHat directory. But the process is very slow even with standard "high-speed" connections. And it's not polite to use this bandwidth from a Red Hat FTP server when others need it as well.

You can also set up your own FTP or HTTP server. If you already have a Linux or Unix computer, you can also set it up with an NFS server. The basic steps are straightforward; all you need is a /RedHat directory in your server tree. Copy the /RPMS and /base subdirectories from the first installation CD. Copy the files from the /RPMS subdirectory on the second and third installation CDs to the /RedHat/RPMS directory that you've created. I cover this process in more detail in Chapter 8, "Administration and Management."



For Red Hat Linux 9, if you're installing over a network and are using anything but the default Red Hat Linux vsFTP server, you'll need to install via an NFS or HTTP share.

Once you've configured and activated your HTTP or FTP server, make sure any firewalls that you might have—such as Microsoft's Internet Connection Firewall or Linux's iptables—aren't blocking communication with that server. Then follow these steps:

1. Boot your computer with the first installation CD or a boot floppy that you may have created earlier in this chapter with the RAWWRITEWIN.EXE utility.
2. When you see the "boot:" prompt, type **text askmethod** and press Enter to run the Red Hat Linux installation program in text mode. This allows you to select a network installation.



Unless you're installing Red Hat Linux 9 through an NFS server, you can install Linux only by using the text-mode interface.

3. You'll see a friendly blue screen where you can select a language for the installation process. Later, you'll get to select languages to install with the operating system.



In text mode, use the Tab and arrow keys to switch between options, and the spacebar or Enter keys to accept a selection.

4. Select a keyboard; if Red Hat Linux has detected your keyboard, it's highlighted here.
5. Choose the installation method. You can install from installation CDs on the local computer, or you can install from NFS, FTP, or HTTP servers on remote computers.
6. If you started the installation process with a boot floppy, Red Hat may prompt for one or both driver floppies. Select the "Use a driver disk" option.
7. Select the drive associated with your driver floppy; fd0 corresponds to the first floppy drive on your computer.
8. Insert the driver disk and continue; Red Hat reads the information on your disk, looking for the network, SCSI, or PCMCIA driver that it might need. Repeat these steps with the other driver disks if prompted.
9. Because you're installing Linux from a network source, the installation program needs to assign you an IP address over the network. If you have a DHCP server for your network (it can even be a Microsoft Windows DHCP server), you can use dynamic IP configuration; otherwise, you'll have to set up the IP address statically.
10. Finally, you'll need the name or IP address of the network server, along with the location of the /RedHat directory. For example, **Figure 8** illustrates an FTP installation.

If you're installing Red Hat Linux from an NFS or HTTP server, the information that you need is essentially the same. Once Red Hat finds the remote server, it reads the installation files and starts the installation process. The information you need in text mode is essentially the same as if you were installing Linux in graphical mode.

Conclusion

This chapter examined the purposes of a file server, to help guide you as you install Linux. While this book focuses on Red Hat Linux, there are a wide variety of different distributions available.

While Red Hat Linux installs on most computers without problems, it's a good idea to collect information about your hardware, especially if you're installing Linux on several computers. There are a number of resources available for Linux hardware support.

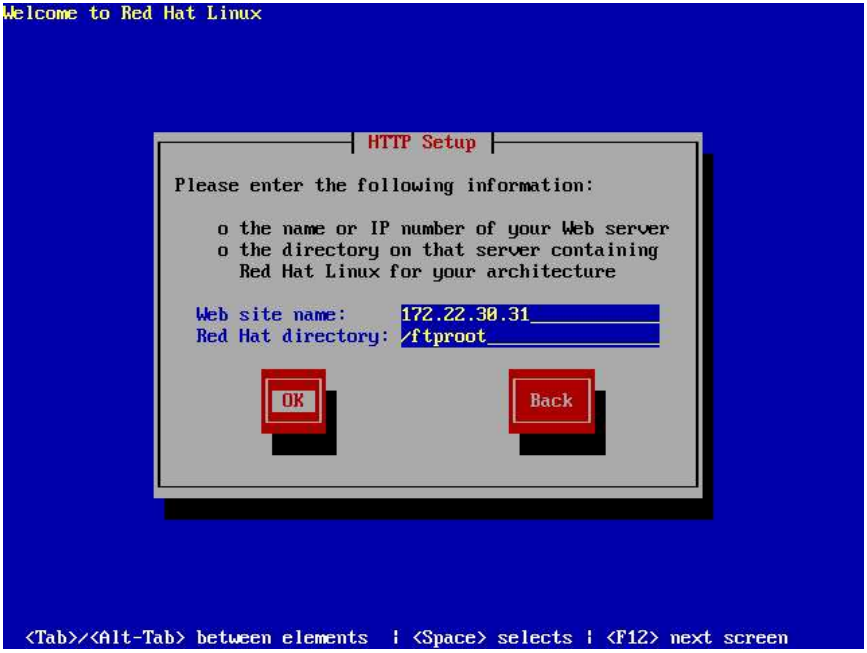


***Figure 8**. Installing Red Hat Linux from a remote FTP server.*

Defining your hardware can help as you define the available space for Linux on your computer's hard drive. There are a number of different ways you can organize Linux on partitions within the Filesystem Hierarchy Standard. You can then configure and format desired partitions with FIPS.EXE, Linux's fdisk, and Red Hat's Disk Druid. These tools can even help you set up Linux in a dual-boot configuration with Microsoft Windows.

Finally, this chapter examines information you need during the first part of the Red Hat Linux installation process—from a local installation CD and also from a remote server. In the next chapter, I take a detailed look at the remainder of the installation process. This will help you install just the software you need to set up a File and Print server on Linux.

> Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 2
Installing Linux as a File Server

When you install any operating system, it's best to install just the software that you need. You don't want to waste space that could be used for real necessities. If you install too much software, you might open up security holes that a cracker could use to break into your system. When you're done with this chapter, you'll know how to install Linux on your computer, with just the software that you need, available when you need it.

This chapter is focused on installing the software you need to set up Red Hat Linux as a file server. I detail the requirements to turn this computer into a print server in Chapter 7, "Configuring Printers." The first part of this chapter is a continuation of Chapter 1, focused on the software selections that you can and should make for your system. It also guides you through the critical choices you can make for firewalls and authentication.

If you install and configure Red Hat Linux to log in at the graphical interface, there is another step after installation is complete. In that case, Red Hat Linux guides you through a configuration process known as First Boot. This works hand in hand with a connection to the Red Hat Network (RHN), where you can make sure that your installed software is kept up to date with the latest security enhancements.

Sometimes you'll want to install or remove additional software. The Red Hat Package Manager utility allows you to interface with the installation CDs or network sources. Many third parties create software that conforms to the Red Hat Package Manager (RPM) system. You need to know a few key commands to make sure that each associated service is active whenever you boot Linux.

When you install Linux on your computer, you get an operating system that's designed for networking. Linux is a clone of Unix, which was designed by many of the same people who developed the public network that eventually became the Internet. The next section includes a number of basic networking theory terms that you need to know. If you're familiar with Microsoft Windows networks, you're probably already familiar with most of the terms.

Basic terms

Authentication: All topics related to user names, passwords, and any associated databases.

DNS: The Domain Name Service is a database of domain names such as **www.hentzenwerke.com** and IP addresses such as 10.11.12.13.

DHCP: A Dynamic Host Configuration Protocol server assigns IP addresses and other network information to different computers, normally during the boot process for the operating system.

Firewall: A software service that looks at all data coming and going from a network; it can be configured to block certain types of data.

FTP: The File Transfer Protocol is optimized to speed network file transfers.

HTTP: The HyperText Transfer Protocol is associated with Web pages.

LAN: A Local Area Network consists of two or more computers connected to communicate with each other.

NFS: The Network File System is the standard way that Linux and Unix computers share directories on a network. NFS usually doesn't work with Microsoft Windows computers.

NIS: The Network Information Service supports sharing authentication and other key files between Linux and Unix computers on a network.

POP3: The Post Office Protocol is associated with incoming e-mail.

SMB: The Server Message Block protocol is a foundation of Microsoft Windows networking. Also known as the Common Internet File System. The Linux implementation of SMB is known as Samba.

SMTP: The Simple Mail Transfer Protocol is associated with outgoing e-mail.

SSH: The Secure Shell allows secure encrypted access over a network.

TCP/IP: This refers to a set of protocols that most computers use to communicate with each other on a network. It's the default for Linux and the Internet.

The installation nitty-gritty

This section is essentially a continuation of Chapter 1, where I explained the basic steps required to install Red Hat Linux. Now I'll cover the remainder of the process, which involves setting up a firewall, designating an authentication server, configuring your graphics hardware, and installing just the software that you need.

Although this section describes the installation process associated with Red Hat Linux 9, you can use the skills described here to install a number of other Linux distributions.

Remaining steps

In Chapter 1, you read about the basic steps required to install Red Hat Linux on your computer, through the point where you're ready to configure network interfaces. I detail many of these steps shortly. The remaining steps are as follows:

1. Select a firewall to configure. You can activate the firewall on the network card of your choice.
2. Choose additional languages to install with Red Hat Linux, as desired. You'll be able to switch between languages in the GUI by using the redhat-config-languages utility.
3. Set a time zone for your computer. If Linux is the only operating system installed, activate the "System clock uses UTC" option. That sets your BIOS clock to Greenwich Mean Time; Linux then sets its clock relative to that time.



If you're installing Linux on the same computer as Microsoft Windows, don't activate the "System clock uses UTC" option, because Microsoft Windows doesn't understand it.

4. Designate a root password, which is the administrative password for this computer. Like the Microsoft Windows administrator password, it's the most important password on your system.
5. Configure authentication for your system. You can set up encryption for your passwords, as well as the use of Linux or Unix or even Microsoft Windows servers.

6. Select the packages to be installed. Later in this chapter, I'll specify the package groups required to create a File and Print server that can interact on a network with Microsoft Windows computers.
7. Red Hat Linux proceeds with installation. Once the desired packages are installed, create a boot floppy customized for your installation.
8. Assuming you've installed a GUI, you can configure the video card associated with your system. In most cases, Red Hat detects your video card and associated settings automatically; you just have to confirm the selection.
9. Select the monitor that is connected to your system. In most cases, Red Hat detects this monitor automatically.
10. Choose a default resolution; your selections depend on the capabilities of your graphics hardware. If you select a Server or Custom installation (see Chapter 1), you'll also be able to choose between text and graphical login screens. For the purposes of this chapter, I'll assume that you've selected a graphical login screen.
11. That's it! Red Hat Linux reboots. If you've selected a graphical login screen, you'll proceed to First Boot, which is described later in this chapter.

Configuring a firewall

To understand how a firewall works, let's step back and take a brief look at Linux networking theory.

The protocols associated with TCP/IP are like channels on a TV. When you set up a firewall, you're blocking many (if not most) of these channels. For your users, this is a form of censorship. For crackers who want to get into your system, it makes life a lot more difficult.

When you set up a Linux computer on a network, the default language is known as TCP/IP. It happens to be the language that computers use on the Internet. It's made up of a large number of protocols, or rules for computer communication. Some examples of TCP/IP protocols are HTTP for Web pages, POP3 for e-mail, and FTP for file transfers. Conceptually, it's like a big two-way TV set, with 65,536 ports that work like two-way TV channels. Unfortunately, crackers can try to break into your system on any of these channels.

When you configure a firewall during the Red Hat Linux installation process, you'll see a screen similar to **Figure 1**.

As you can see, there are three basic security levels associated with Red Hat Linux firewalls:

- **High**: A high-security firewall blocks most incoming traffic to your network, except the traffic that comes from DNS servers. A DNS server is most often used to translate domain names such as **www.mommabears.com** to IP addresses such as 192.168.0.33. You can still get data that you might request from an outside network, such as Internet Web pages.
- **Medium**: A medium-security firewall blocks incoming traffic over the most commonly used TCP/IP channels—those between 1 and 1023. It also blocks communication on NFS and remote GUI connections.
- **No firewall**: You can disable firewalls on the local computer. This is appropriate for a computer that's inside a LAN that is already protected by a firewall.

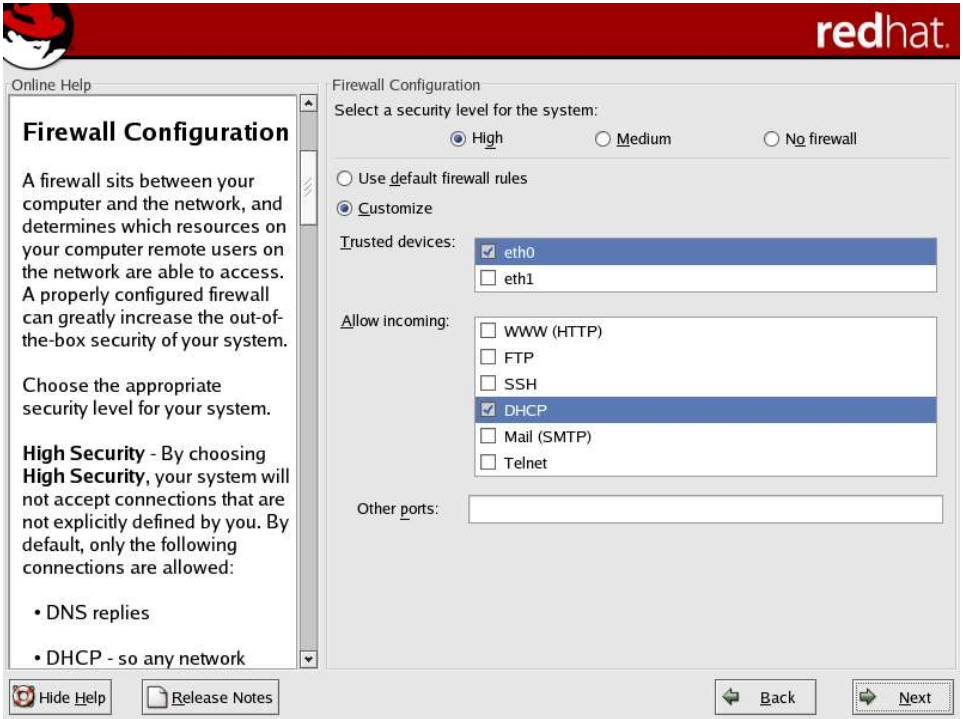


***Figure 1**. Configuring a firewall during Red Hat installation.*

Once you understand the types of firewalls available, you can reconfigure the firewall after Linux is installed. Red Hat Linux includes a couple of good tools that help with this process. The commands used to launch these tools are **lokkit** and **redhat-config-firewall**. The interfaces are different, but the information that you need for each tool is essentially the same.

Configuring communication through the firewall

Sometimes you want to do more than just browse the Internet from your computer. You may want to set up a Web server. And yes, you may even want to share files. As you can see from Figure 1, Red Hat provides a set of standard services that you can configure in the firewall.

Trusted devices: Some computers, known as gateways, sit between two different networks such as a LAN and the Internet. I have two network cards on my gateway computer. One network card, known as eth0, is connected to my LAN. The other network card, known as eth1, is connected to the Internet. Because I trust the data from my LAN, I've set eth0 as a "trusted device." The firewall defined in Figure 1 is applied only to traffic coming from the Internet, which protects my LAN.

Allow incoming: As shown in Figure 1, there are a number of default services that you can let through the firewall. These are needed only for inbound requests. For example, you don't need to activate the WWW (HTTP) box to allow you (or users on your network) to browse the Internet. Here are the default services:

- **WWW (HTTP)**: Allows users who are outside the firewall to access a Web server on your computer.
- **FTP**: Permits external users to access an FTP server on your computer.
- **SSH**: Allows users outside your firewall to access your computer with the Secure Shell (SSH).
- **DHCP**: Allows a DHCP server outside your firewall to provide IP address information to computers behind the firewall.
- **Mail (SMTP)**: Permits computers outside your firewall to use a mail server on your computer.
- **Telnet**: Allows users outside your firewall to access your computer with the Telnet service. This is not secure; SSH is the preferred alternative.

Authentication

If you've selected a Server or Custom configuration for Red Hat Linux, you're setting up a computer that can hold user names and passwords for your network—or at least direct requests to the responsible servers. In those cases, you'll need to configure authentication as shown in **Figure 2**.



If you're installing Red Hat Linux in a Personal Desktop or Workstation configuration, you're configuring a computer that isn't designed to hold or handle group or Domain user names or passwords, so you won't see Figure 2.

You can set up authentication in a number of ways during the Red Hat Linux installation process. First, by default, Red Hat supports MD5 passwords of up to 256 characters. It also sets up shadow passwords that are encrypted in a file accessible only to the root user (see Chapter 3, "Setting Up Your Server File System"). As you can see, the authentication screen includes tabs for NIS, LDAP, and Kerberos 5. I'll cover these briefly, but these topics are beyond the scope of this book:

- **NIS**: The Network Information System can provide a common authentication database for Linux and Unix computers. You can configure NIS to store common key files to be used by all Linux and Unix computers on the network.
- **LDAP**: The Lightweight Directory Assistance Protocol is similar in functionality to Linux and Unix computers. Unfortunately, at the time of this writing, the latest stable version of Samba did not support LDAP, which is the main reason why you can't yet configure a Linux computer as a Domain Controller on a Windows 2000 native mode network. I believe this problem will be addressed when Samba 3.0 is released.
- **Kerberos 5:** The Kerberos system, developed at MIT, is designed to eliminate the need to send passwords over a network. Unfortunately, it does not work with the default shadow passwords described earlier, or other Linux authentication systems.

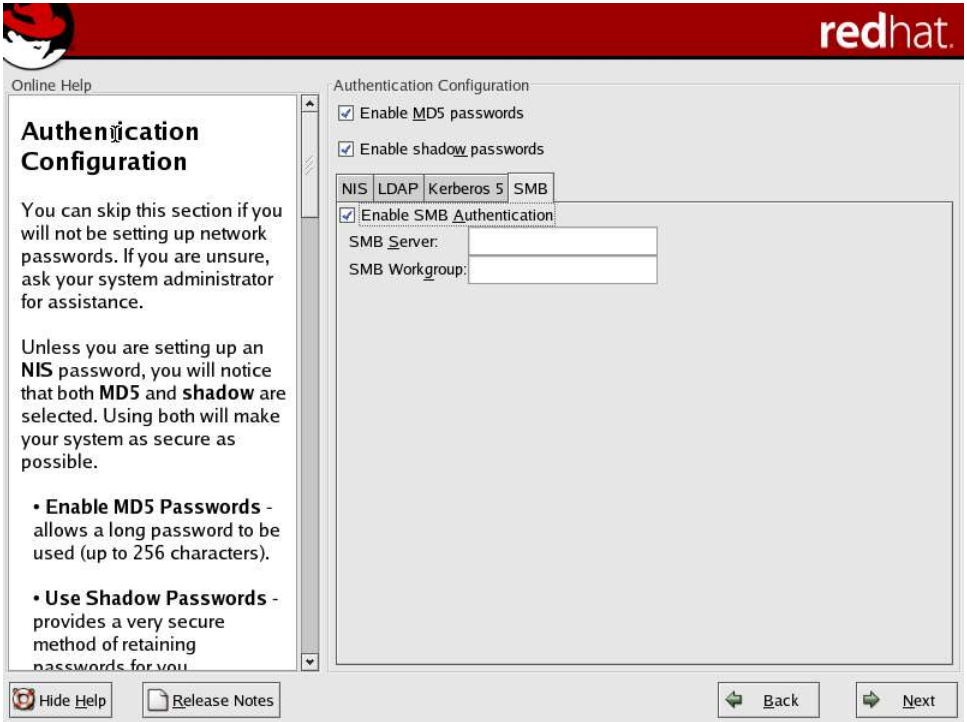


***Figure 2**. Setting up authentication during Red Hat installation.*

Samba authentication

You can configure Samba authentication on the SMB tab in Figure 2. This requires installation of the Windows File Server package group described later in this chapter. If you're not yet familiar with Samba, feel free to skip this section. You can configure all of the settings described on the SMB tab in Chapter 3, "Setting Up Your Server File System."

Despite the reference to "SMB Workgroup," you can configure a Samba server to refer login requests to the Primary Domain Controller (PDC) on a Microsoft Domain. Here are the three options on the tab:

- **Enable SMB Authentication**: Allows you to set this Linux computer to view the user name and password database on a Microsoft Windows Domain.
- **SMB Server**: Allows you to list the name of a PDC on a Microsoft Windows Domain.
- **SMB Workgroup**: Despite the name, this permits you to name the Microsoft Windows *Domain* to which you want this computer to connect. As you'll see in Chapter 3, the Samba **workgroup** variable can be used to name the Workgroup or Domain that you want your computer to join.

Next comes the nitty-gritty of the installation process: selecting the software that you want to install with Red Hat Linux.

Installing what's necessary

During the Red Hat Linux installation process, you can select the package groups that you want to install on your system. These groups install everything from a GUI to database servers.

If you have a relatively modern hard drive, chances are good that you have plenty of space. You could install all the software available on Red Hat Linux and still have more than enough space for data. But as described earlier, that's not the best idea for security reasons.

In the following sections, I explain what you can install with Red Hat Linux—the basic components, as well as the services that you need for a File and Print server, and other network services.

Basic components

If you have a small hard drive, or if you're very concerned with security, you may be tempted to install just the minimum software required for the Linux operating system. The minimum space required by Red Hat Linux is less than 500MB, which sounds appealing relative to the 2GB of software installed on my Microsoft Windows XP Professional C:\Windows directory.

In fact, a minimum installation of Red Hat Linux supports networking with other Linux and Unix computers with NFS. It includes the basic tools and commands that make Linux so robust. You can even connect such a system to the Internet. But for those of you who are converting from Microsoft Windows, it is probably not enough. A minimum installation does not include a GUI or the tools required to share files or printers with Microsoft Windows computers on your network.

Basic components plus

To take full advantage of what Linux can do for you, it is best to learn the command-line interface. Text commands may seem cryptic at first, but they are often the quickest and most efficient way to administer a computer. However, command-line skills are no longer required to configure a Linux computer or server on a Microsoft Windows network. Later in this chapter I'll describe some tips and tricks that you can perform at the command line.

Shortly, I'll illustrate a fairly minimal configuration with a GUI and services that support a File and Print server. This configuration installs about 1.5GB of files on your computer.

Options

A number of optional package groups are associated with Red Hat Linux. A package group is just that—a group of common software packages. For example, two different print systems are part of the Printing Support package group; these are known as CUPS and LPD and are described in Chapter 7, "Configuring Printers."

When you install Red Hat Linux in the standard graphical mode, you'll eventually see a screen similar to **Figure 3**, which illustrates the package groups that you can install.

In Figure 3, I've selected the X Window System package group. When you select this package group for installation, 31 of 33 associated packages are installed by default. If you click the Details link, you can select or deselect some of these packages. The same pattern holds true for the other package groups in this window.

The different package groups are summarized in **Table 1**, and are in the same order as you might find during the Red Hat Linux 9 graphical installation process. I won't install most of these for the purposes of this book, but you can select some of these groups to configure your computer for more than just file and print services.

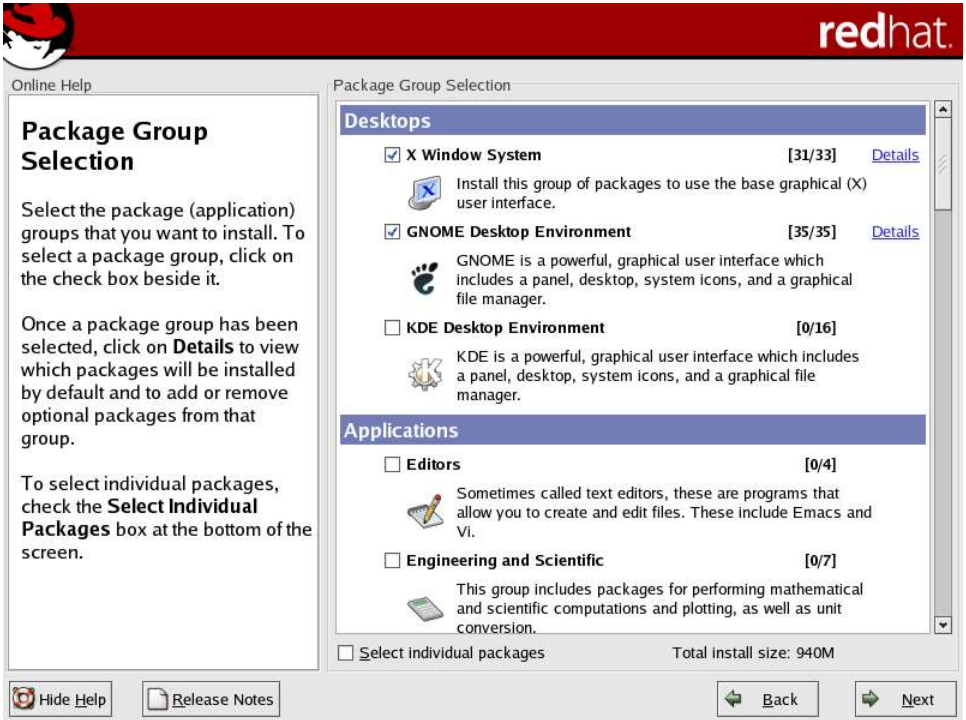


Figure 3. *Selecting package groups during Red Hat Linux installation.*

Table 1*. Red Hat Linux package groups.*

Package Group	Contents
X Window System	The base packages include graphical software, fonts, and configuration files. Additional packages include related administration and configuration utilities.
GNOME Desktop Environment	GNOME base packages include a GUI window and session manager as well as some of the basic utilities that you might expect in any GUI. Additional software includes small-scale GUI programs such as an FTP client. GNOME is the default GUI for Red Hat Linux.
KDE Desktop Environment	KDE is the main alternative to GNOME, and follows the same basic pattern for base and additional packages. KDE is the default GUI for many of the other major Linux distributions.
Editors	Many, perhaps most, Linux administrators prefer to configure Linux by editing configuration files in text editors. The Editors package group includes the popular emacs text editor, as well as enhancements for the more fundamental vi text editor. I explain vi in Chapter 8, “Administration and Management.”
Engineering and Scientific	Several engineering and scientific tools can be installed as optional components with Red Hat Linux.
Graphical Internet	You can install a number of different graphical browsers, e-mail, chat, and videoconferencing applications.

Package Group	Contents
Text-based Internet	If your computer can't handle graphical interfaces, you can install a number of text-based browsers, e-mail readers, chat, and FTP clients.
Office/Productivity	While Red Hat Linux normally comes with the OpenOffice.org suite, you can install the components of two other office suites, as well as several additional graphics packages.
Sound and Video	With the components of this package group, you can configure CD players, sound synthesizers, CD and DVD recorders, and more.
Authoring and Publishing	This package group includes authoring tools closely associated with Linux and Unix, such as TeX and docbook.
Graphics	A number of different Linux-compatible applications can help you create and manipulate graphical images.
Games and Entertainment	Linux can be installed with a wide variety of computer games.
Server Configuration Tools	A key group that includes a wide variety of graphical administration tools.
Web Server	Linux is the platform associated with the most popular Web server on the Internet, Apache. You can install Apache, associated modules, and a kernel-based Web server known as the Red Hat Content Accelerator or Tux.
Mail Server	Linux is also a popular platform for mail services such as sendmail.
Windows File Server	The key group for this book. Allows you to configure your Linux computer to look like a server on a Microsoft Windows network.
DNS Name Server	It's not difficult to set up a DNS server on your Linux computer for your network.
FTP Server	If you want to share files with clients that use an FTP interface, this package installs the Red Hat very secure FTP daemon, vsFTP.
SQL Database Server	If you're managing databases, you may already be using one based on the Structured Query Language (SQL). This group can install packages associated with the PostgreSQL and MySQL database systems.
News Server	Red Hat Linux can be configured with a news server.
Network Servers	Linux can be configured with lots more than just Windows file servers; this package includes servers for DHCP, RSH, NIS, Telnet, and more.
Development Tools	If you ever get into the nitty-gritty of building Linux software or reconfiguring the Linux kernel, you'll need some of the tools in this package.
Kernel Development	Several special packages are needed to reconfigure and then recompile the Linux kernel for different settings.
X Software Development	If you ever want to develop graphical software or associated fonts, you'll need the software libraries from this package group.
GNOME Software Development	If you ever get into developing some of the applications associated with the GNOME desktop environment, you'll need the libraries from this group.
KDE Software Development	If you ever want to develop some of the applications associated with the KDE desktop environment, you'll need the libraries from this package group.
Administration Tools	A key group. This includes a number of graphical tools that you can use to configure your Linux computer.
System Tools	A diverse variety of tools. They include clients for some of the software in the Network Servers package group. You'll need at least the **samba-client** package to connect to a Microsoft Windows network.
Printing Support	Red Hat Linux allows you to install two different print systems, which are described in more detail in Chapter 7, "Configuring Printers."

You can select the packages of your choice during the installation process.

Basic components: This is a minimal installation. Deselect all packages shown in the Package Group Selection screen (Figure 3). Even when you deselect all package groups, Red Hat Linux still installs the basic components required for the operating system, including components that support basic networking. However, this minimal installation does not include a GUI and does not support connections to a Microsoft Windows network.

Basic components plus: If you want a Linux file server that can work on a Microsoft Windows network, you need to add the following package groups (the default packages within each group are sufficient):

- **Windows File Server**: As described in the table, the Windows File Server packages allow you to connect your computer as a server, even a PDC, on a Microsoft Windows network.
- **System Tools**: There is one required package in this group, **samba-client**. It allows you to connect your computer as a client in a Microsoft Windows-style network.
- **Printing Support**: The default print service associated with Red Hat Linux is CUPS; while LPD is a part of this package group, you don't need it, and it isn't installed by default.

Because this is a book for users converting from Microsoft Windows, I've also installed the following package groups. I refer to the software in these groups throughout the rest of the book.

- **X Window System**: This group includes the foundation packages of the Linux GUI.
- **GNOME Desktop Environment**: GNOME is the default GUI for Red Hat Linux, and supports the use of the GUI tools described in later chapters. If you prefer, you can substitute or add the KDE Desktop Environment.
- **Graphical Internet**: Linux includes a number of GUI applications such as Web browsers that are part of this group. A Web browser is required for the Samba Web Administration Tool (SWAT) described in Chapter 3, "Setting Up Your Server File System."
- **Server Configuration Tools**: Red Hat Linux has developed a number of GUI tools for various services (such as Samba) for sharing with Microsoft Windows networks and print managers.
- **Administration Tools**: Red Hat has developed a group of basic GUI administrative tools that can help you configure a computer, manage users, review log files, and more.

I've installed several other package groups so I can use this Linux server as a complete workstation. If you're just learning Linux, you should consider installing the following package groups:

- **Office/Productivity**: By default, Red Hat Linux installs the OpenOffice.org suite of office applications. You can open and save files in Microsoft Office format. While it isn't 100 percent compatible with Microsoft Office, it's sufficient for most uses. In fact, I've written all of this book using OpenOffice.org Write.

- **Sound and Video**: Red Hat Linux includes a number of utilities that can help you mix different sounds, play music, and record sound and data—on both CDs and DVDs.
- **Graphics**: Red Hat Linux includes a number of applications that many believe are at least as capable as commercial programs such as Paint Shop Pro. For example, a number of the screenshots in this book were created using The GIMP (also known as the GNU Image Manipulation Program).

The default packages associated with each selected group work well for the purposes of this book; for the package groups that I've listed, Red Hat installs 1.5GB of files by default.

Other network services

There are several other network services that you'll find useful when you install them with Red Hat Linux. You can install them from the Package Group Selection window (see Figure 3). These packages are part of the various server package groups.

If you don't need these services, don't install them. If you do, they may provide a conduit for a cracker to get into your computer. You can always install these services later. Please note that additional information on these services is beyond the scope of this book.

These services include:

Amanda: The Advanced Maryland Automatic Network Disk Archiver supports regularly scheduled backups from a Linux server.

Apache: The Apache Web server is by far the most popular Web server on the Internet. It allows you to set up secure Web sites, databases, scripts, proxies, and more.



For more information on Apache from the viewpoint of a Windows administrator, see another book in this series, Linux Transfer for Windows Web Server Admins, 2003.

bind: The Berkeley Internet Name Domain (bind) provides the software that allows Linux to act as a DNS server. Also known as a nameserver.

caching-nameserver: A caching nameserver that can store recent requests from remote DNS servers.

CIPE: Crypto IP Encapsulation provides secure connections between private networks through insecure networks such as the Internet. Also known as Virtual Private Networking.

Finger: Administrative tool that supports queries about users on local or remote computers.

MySQL:An open-source database-management system. Can be used with Apache for Web applications.

InterNetNews: A news server commonly used on Usenet, the bulletin board system associated with Internet message groups.

NIS: The Network Information Service provides a common database of authentication files and more, as described earlier.

Postfix: A common alternative to sendmail, which you can also configure as a service for incoming and outgoing e-mail.

postgreSQL: An open-source database-management system. Can be used with Apache for Web applications.

PXE: Supports a Pre-boot eXecution Environment for remote Linux computers to boot from a disk image.

RSH: The remote shell provides a method for remote users to connect to the local command-line interface. Not secure.

sendmail: A popular service for incoming and outgoing e-mail. If you set up a sendmail server, you can then point e-mail clients such as Mozilla, Evolution, and even Microsoft Outlook to the sendmail mail server.

SSH: The Secure Shell allows encrypted remote logins, unlike Telnet.

Telnet: The Telnet server supports remote login to the local command-line interface. Not secure.

Tux: An alternative Web server to Apache; provides faster service for certain types of files such as pictures. Can be configured to work hand-in-hand with Apache for different types of files.

VNC: When you set up a Virtual Network Computing server, remote users can connect to and control your GUI.

vsFTP:The very secure FTP server allows you to set up file services to real and anonymous users. When you connect to **ftp.redhat.com**, you're connecting to a vsFTP server.

To review, once you've made your software selections, Red Hat installs the software on your computer. You can then create a boot disk, configure your graphics hardware, and choose a graphical or text login screen. For the purposes of this chapter, I'm assuming that you've selected a graphical login screen. Red Hat Linux then reboots your computer.



If you installed Red Hat Linux in the Personal Desktop or Workstation configuration, Red Hat doesn't give you a choice. It'll start with a graphical login screen automatically.

After installation

Once Red Hat Linux is installed, there are a few more things to configure. *Before* you log in to Linux for the first time, Red Hat takes you through the First Boot process. Once you've set up basic parameters with First Boot, you can keep your computer up to date—with a connection to the Red Hat Network.



The steps you take in this section depend on whether you've selected a graphical or text login screen.

If you don't connect to the Red Hat Network, you can get the latest Red Hat packages through what is known as Rawhide. I'll cover this later in the chapter. And there are other download sites that you can use to keep the software on your computer up to date.

First Boot

When Linux boots on your computer, you should see the GRUB boot loader described in Chapter 1. This is the default, and should automatically boot Linux 10 seconds after you see the GRUB menu. Linux then starts the boot process, during which it sets up connections to your hardware, looks for new hardware, and then starts the First Boot process, which includes the following steps:

1. Review the welcome screen. On the left-hand pane, you can review your status in the First Boot process. See **Figure 4**. You may not see all the steps shown here. For example, if you don't have a sound card, you won't see a Sound Card entry. Click Forward to continue.

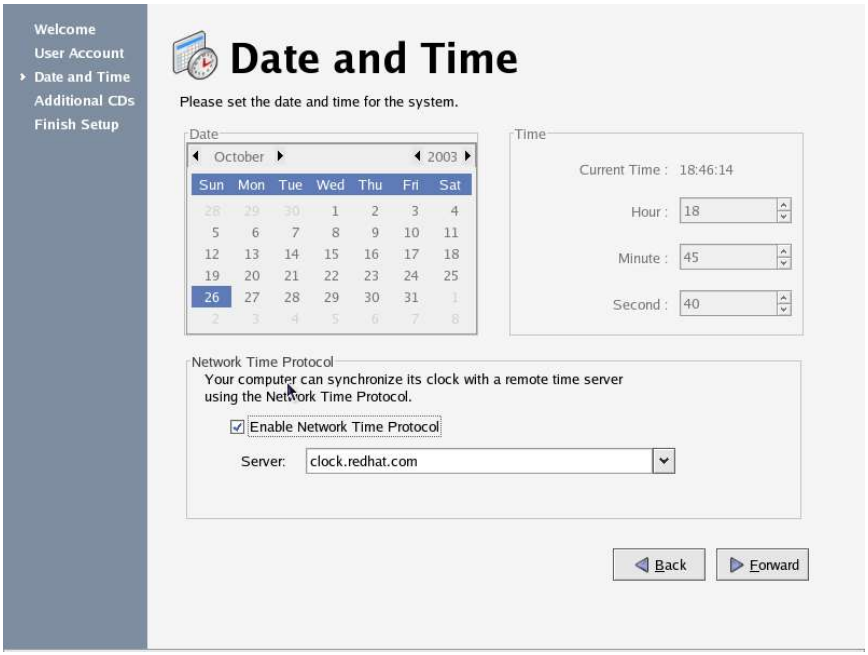


***Figure 4**. Continuing the First Boot process. You can follow your progress in the left-hand pane, and set the date and time in the right-hand pane.*

2. Create a regular user account. This account won't have administrative privileges. If the user name is **mj**, the associated home directory is /home/mj. The text boxes in this screen are almost self-explanatory; enter the desired user name and password in the associated text boxes. In the Full Name text box, enter the information of your choice; it's essentially a comment in the Linux password database. Typing in the full name of the user is a good idea. Click Forward to create that user and continue.



*Even if you're the only person using this computer, you should create a regular user account. Don't log in as root unless you're actually administering the computer. Many Linux gurus assume administrative privileges temporarily by using the **su** command and the root user's password.*

3. Set the date and time on your computer, or use the Network Time Protocol (NTP) to synchronize your computer clock with a central time server, as shown in Figure 4. You can set up a local time server on your network, or you can enter one of the time servers on the Internet. Red Hat provides two of them, as shown in the drop-down

box. These are **clock.redhat.com** and **clock1.redhat.com**. A list of the public Internet time servers is available at **www.eecis.udel.edu/~mills/ntp/servers.html**. Once you've made your selections, click Forward to continue.



*If you don't yet have a network connection to the Internet, you can set up synchronization later by using the **redhat-config-time** command.*



Synchronizing the time on different computers on your network is important; otherwise, your network may not save the most recent revision of your files.

4. If Red Hat Linux detects a sound card, First Boot lists its make and model. If you see a "Play test sound" button, connect your speakers, click that button, confirm if you hear a sound, and click Forward to continue.
5. Now you can register your computer on the Red Hat Network (RHN). This is a good idea, but say no for now. In the next section, you can still register your computer on the RHN. Click Forward to continue.
6. Next, you can install Red Hat Documentation, additional packages from the Red Hat Linux installation CDs, and applications from any additional Red Hat CDs that you may have. If desired, insert the appropriate CD and click the associated Install button. When you're ready, click Forward to continue and complete the process.
7. Assuming you set your computer to log in graphically, you now get to log in to Linux for the first time in a screen similar to **Figure 5**.

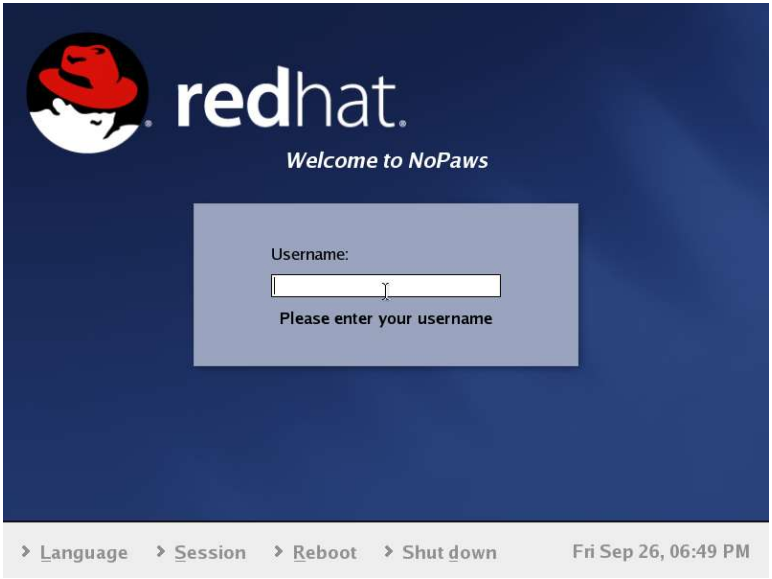


***Figure 5**. The Linux GUI login screen.*



As described earlier, this section assumes that you've selected a graphical login screen during the installation process. If you've selected a text login screen, I assume that you know something about Linux and know a bit about the command-line interface.

*So here's the short version of how to get to First Boot from the text login screen: Log in as the root user, run the **init 5** command, log in at the graphical screen to start the GUI, and run the **firstboot** command from a command-line window inside the Linux GUI.*

Now that you've configured your Linux computer, it's a good time to check the status of your software through the Red Hat Network. You can do so using the Red Hat Linux GUI.

The Red Hat Linux GUI

The default Red Hat Linux GUI is known as GNOME, the GNU Network Object Model Environment. Conceptually, many of the interfaces are similar to Microsoft Windows. As you can see in **Figure 6**, GNOME includes a number of icons on a desktop, and a taskbar (called the "panel" in Linux).

There are three icons on the desktop. In the upper-left corner, the "root's Home" icon can bring you to the directory of the current user—in this case, root. The "Start Here" icon provides an interface to applications and preferences similar to the Microsoft Windows Explorer interface. The "Trash" icon is self explanatory; it works like a Recycle Bin for files that you might delete from within the GUI.

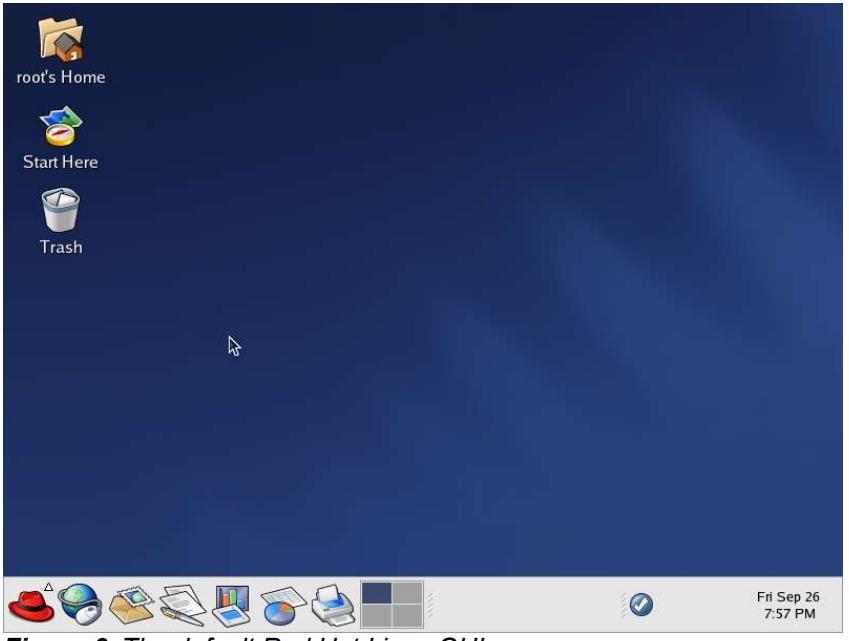


Figure 6. *The default Red Hat Linux GUI.*



If you've "mounted" a CD or floppy in their respective drives, you'll also see icons that allow you to access their files on your desktop. If you double-click one of these icons, this opens the drives in the graphical Nautilus interface, which looks like a simplified version of Windows Explorer.

The picture of a red hat in the lower-left corner of the desktop works like the Microsoft Windows Start button. In Red Hat Linux, it's known as the Main Menu button. You can use it to access a number of programs and utilities. Try it out.

One important Linux skill in the GUI is access to the command-line interface. With your mouse, right-click on the desktop. In the pop-up menu that appears, select New Terminal. In Red Hat Linux, this starts a command-line window, where you can enter regular commands, as well as commands that open various GUI utilities and applications.

The Red Hat Network

The Red Hat Network (RHN) provides a connection between your computer and update servers at **rhn.redhat.com**. It is conceptually similar to the Microsoft Windows Update utility, in that it checks your system against the latest software associated with the operating system.

Naturally, to connect to the RHN, you need a connection between your computer and the Internet. If your computer is on a LAN that's already connected to the Internet, Red Hat should have already configured your computer as required during the installation process.

The RHN is different, because it provides updates for the full range of Linux software associated with the Red Hat Linux distribution. For example, it supports regular updates of key applications such as the Evolution mail manager. However, the RHN also requires that you register your system.

To start the registration process, open the Red Hat Network utility by clicking Main Menu | System Tools | Red Hat Network. The first time you run this utility, you should see the Red Hat Network Configuration window in **Figure 7**.

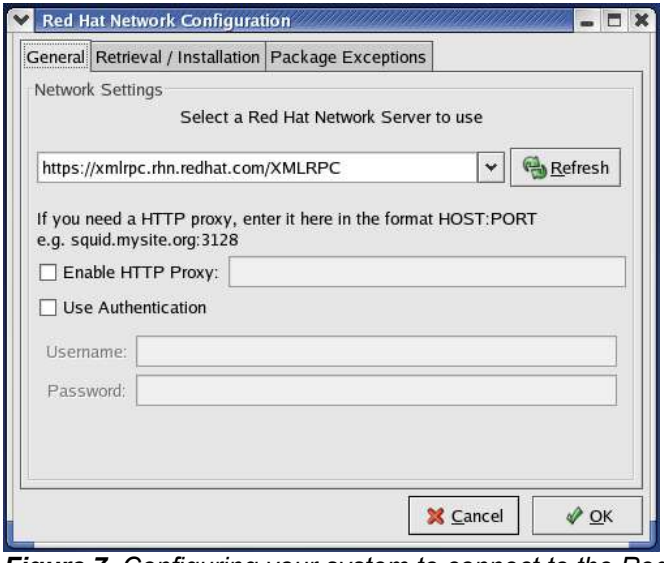


***Figure 7**. Configuring your system to connect to the Red Hat Network.*

This window includes three tabs:

- **General**: To connect to the RHN, you need to specify a server. The default is usually appropriate. If there's a proxy server on your network, you'll need to enter the settings required to connect through that server.



*As Red Hat moves toward the Fedora Project, the procedures associated with the Red Hat Network may change. Monitor **fedora.redhat.com** and **rhn.redhat.com** for more information.*

- **Retrieval/Installation**: The settings on this tab allow you to specify when the package is to be installed, and whether you want to use PGP to verify the integrity of downloaded packages.



*The Linux implementation of Pretty Good Privacy (PGP) is known as the GNU Privacy Guard (GPG). For more information, see **www.gnupg.org**.*

- **Package Exceptions**: You can specify packages that are not upgraded automatically. For example, updated kernels normally are not installed.

The default settings for these tabs should meet the needs of most users. When you're satisfied with your settings, click OK to continue. Now follow these steps:

1. Assuming that you've accepted the default GPG options, you're asked if you want to install a GPG keyring for verifying downloaded Red Hat packages. It's an excellent idea. Click Yes to continue.
2. At the Red Hat Update Agent screen, click Forward.
3. Your computer connects to the Red Hat Network. The first time you start the Update Agent, you'll get to review the current privacy statement. Read it, and assuming you accept, click Forward to continue.
4. The Red Hat Update Agent takes you to the login screen shown in **Figure 8**. As you can see, you have two options for connecting to the RHN. If you have an existing account and an available "entitlement," log in with your current RHN user name and password. If you haven't registered with the RHN before, select Create New Account to set up a *Demo* account for this computer.



*While Demo accounts are free, you can also purchase additional entitlements on the Red Hat Network with various levels of support. There is a limit of one Demo account per person, and each account must be renewed monthly. For more information, see **rhn.redhat.com**.*

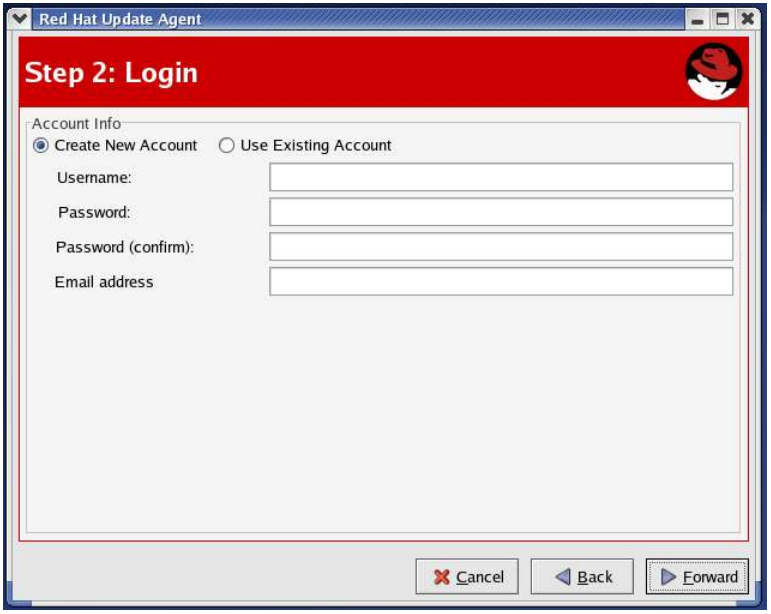


***Figure 8**. Logging in to the Red Hat Update Agent.*

5. The next screen summarizes some basic settings on your computer, including the Profile name (which is by default your computer's hostname), CPU, RAM, IP address, and version of Red Hat Linux. You can change the Profile name, and choose whether to send this basic information to the RHN. Make any desired changes and click Forward to continue.
6. The Red Hat Update Agent collects a current list of installed packages. All of them are listed on the associated screen. By default, all are sent to the RHN; you can deselect the packages of your choice. Click Forward to continue.
7. You're given one last chance to cancel before your profile is forwarded to the RHN. Assuming you're ready, click Forward to continue.
8. Assuming the Update Agent transmitted your profile, you should see a screen entitled "Channels," which should correspond to the version of Red Hat Linux that's installed on your computer. Verify and click Forward to continue. The Update Agent builds a list of updated RPMs on your computer.
9. Depending on what you have installed, the Update Agent checks your packages against the latest updates on the RHN. It lists newer packages on the Available Package Updates screen. An example of this from my computer is shown in **Figure 9**.
10. Select the packages you want to update. When you highlight a package, you can read its description in the lower pane. If there's a security or other type of advisory associated with the package, you can read it when you click View Advisory.

 The packages shown in Figure 9 are particularly interesting for this book, because Samba is the software that allows Linux computers to connect to and become servers

for a Microsoft Windows network. If you've already configured Samba, you'll want to save at least the associated configuration files before accepting the update. Ideally, you should test the updated software with your current configuration files on a test computer, to see if the changes impact your network in any way. For example, if you've updated the Samba software, you'll want to restart the Samba daemons and test their performance as described in following chapters.

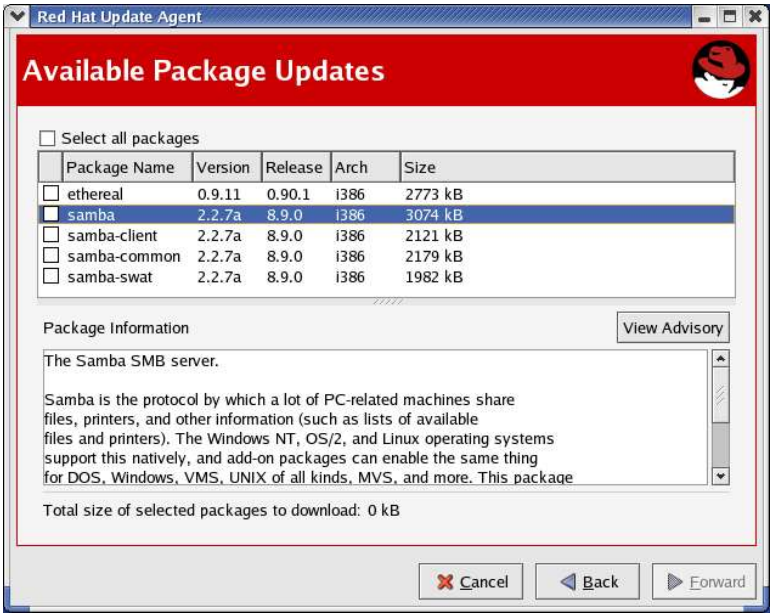


***Figure 9**. The Update Agent lists packages on your system that can be updated.*

11. For the purposes of this chapter, I've accepted all of the proposed changes. Make your selections and click Forward to continue.
12. The Update Agent checks your selected packages for dependencies; any additional packages are added to the update. Next, the packages that you've selected, plus dependencies, are downloaded from the RHN. Once all packages are downloaded, this is the last chance to cancel the update. If you're ready to install the selected packages, click Forward to continue.
13. The Update Agent proceeds to the downloaded packages. When the installation is complete, you'll see the following message: "All finished." Click Forward to continue. When you do, you'll see a summary of the updated packages.



Once you've set up the Update Agent, you can use the Red Hat Network Alert icon. If it's a red exclamation point, updated software is available from the Red Hat Network. If it's a green circle with arrows, your computer is currently checking your software against what's available in the RHN. If it's a blue checkmark, the software on your system is up to date.

There's another way to update the package groups currently installed on your Linux computer: You can use Rawhide.

Rawhide

While the latest versions of software packages should be available through the RHN, the list you see is not complete. For example, some of the software available through the Update Agent may only address security issues, while Rawhide may include upgrades to the actual software.

You can check for the latest versions of software associated with Red Hat Linux by using the Rawhide directory on a Red Hat FTP server or associated mirror sites. On the Red Hat FTP server, you can find Rawhide files in the /pub/redhat/linux/rawhide directory. Different subdirectories are associated with different CPUs; you can download Intel 32-bit compatible packages from the i386 subdirectory.

Other updates

Especially when it comes to security, Red Hat is conscientious about updating packages and making them available through the Update Agent and Rawhide on a timely basis. However, if you're extra sensitive about security, you can go to the source for security updates, the SysAdmin, Audit, Network, Security (SANS) Institute at **www.sans.org**.

It's helpful to become familiar with the organizations behind the software you need to create a Linux File and Print server on a Microsoft Windows network. The basic software, Samba, is developed primarily by a worldwide group of volunteers. Their work is documented at **www.samba.org**.

The default print service for Red Hat Linux, as described earlier, is CUPS. More information and the latest updates for CUPS are available from its sponsor, Easy Software Products, at **www.cups.org**.



The versions of CUPS and Samba included with Red Hat Linux are both covered under the same license as the Linux operating system, the General Public License (GPL). More information and a copy of this license are available from the Free Software Foundation at ***www.gnu.org/copyleft/gpl.html****.*

Package management

Once Linux is installed on your computer, there are other ways to keep your system up to date. Distributions such as Red Hat Linux are built upon the Red Hat Package Manager (RPM) system. An RPM is an integrated group of files. When opened with the right **rpm** command, it installs or upgrades a specific software component. Alternatively, you can install an RPM in the Linux GUI.

To maintain Red Hat Linux, you need to use more than just the Red Hat Update Agent. There are times when you'll need to install or upgrade a specific software package from the command line. To this end, it's important to take a detailed look at the **rpm** command.

Installing and upgrading RPMs

When you installed Red Hat Linux on your computer, you actually installed a series of RPM packages. There are more than 1,400 RPMs on the three Red Hat Linux 9 installation CDs. You can see a list of installed RPMs in the /var/log/rpmpkgs file.



Most Linux files are text files, which you can open in a text editor. In the Red Hat Linux 9 GUI, you can select a text editor by clicking the Main Menu button (the red hat) in the lower-left corner of the desktop. This opens the GNOME Main Menu, from where you can select Accessories | Text Editor, which opens an editor functionally similar to Microsoft Windows' Notepad.

While you're in the GNOME GUI, it's easy to install or remove additional packages from the Red Hat Linux CDs. Insert the first CD in the appropriate drive. Accept the **autorun** option when prompted. This should start the Package Management utility. Click Forward to continue.



*If **autorun** doesn't automatically start on your computer, click Main Menu | System Settings | Add/Remove Applications, or run the **redhat-config-packages** command from a command-line window inside the GUI.*

The Red Hat Linux Package Manager reviews the list of installed RPMs from /var/log/rpmpkgs. This opens the Add or Remove Packages screen shown in **Figure 10**, which should look familiar. It contains essentially the same information and interface that you saw during the Red Hat Linux installation process (see Figure 3).

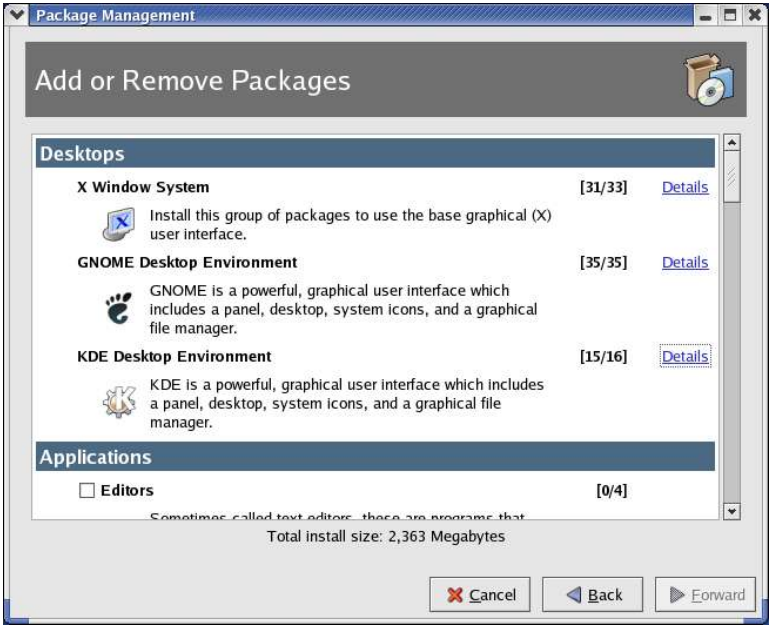


***Figure 10**. The Red Hat Linux Package Management utility.*

If you haven't selected a package group before, you can select it for installation here. For example, you can select the Editors package group as shown in Figure 10. For currently installed package groups, if you click Details you can select or deselect any packages that are not "mandatory." When you're finished, click Forward and follow the prompts. Package

Management checks selected RPMs for dependencies and adds those packages to the list to be installed.

If you want to install or remove any RPMs that are not on the list, you'll need to know a few **rpm** commands. Open a command-line interface in the GUI as described earlier. Right-click on the desktop and select New Terminal.

Using the rpm command

Red Hat RPM packages normally end with the .rpm extension. One useful RPM to install from the installation CD is **rpmdb-redhat**. It can help you identify the source RPM for any file that you have not yet installed. In Red Hat Linux 9, the full name of the package is **rpmdb-redhat-9-0.20030313.i386.rpm**. You can find it on the third Red Hat Linux 9 installation CD.

To install this particular package, follow these steps:

1. Insert the appropriate Red Hat Linux installation CD. For Red Hat Linux 9, that happens to be the third CD. If you're not sure, some trial and error may be required. If you're working in the GNOME desktop, Red Hat should mount the CD automatically. If it does, you'll see a Nautilus window with the files on the main directory of the CD.
2. Open a command-line interface. Right-click on the desktop. In the pop-up menu that appears, select New Terminal. This should open a terminal with a command-line interface inside GNOME.
3. If you don't see a Nautilus window with the CD's files, you'll need to mount the CD. If you do see the Nautilus window, skip to step 6.
4. At the command-line interface, check the account. If you're not already logged in as the root (administrative) user, do so now with the following command (and use the root user's password).

   ```
   # su
   Password:
   ```

5. Now you can mount the CD on the drive. This command mounts the files from the CD on the /mnt/cdrom directory:

   ```
   # mount /mnt/cdrom
   ```

6. You should now have a mounted Red Hat installation CD, from where you can install any desired RPMs. The RPMs in any Red Hat installation CD are located in the RedHat/RPMS subdirectory. I explain this command in more detail in the next section. For now, to install the aforementioned package, run the following command:

   ```
   # rpm -Uvh /mnt/cdrom/RedHat/RPMS/rpmdb-redhat-9-*.i386.rpm
   ```



There's one thing you can do to make commands easier: Use the Tab key. For example, assume that you've typed the first part of this command:

```
# rpm -Uvh /mnt/cdrom/RedHat/RPMS/rpmdb
```

*Pressing the Tab key completes the command with the full name of the package. If more than one package starts with **rpmdb**, press the Tab key twice. You'll see all available options.*

7. The **rpm** command checks your system for previous versions of the noted package. The package is installed or upgraded as needed.

rpm command details

As of this writing, you can't run all administrative functions from the GUI. Sometimes, you'll just need to use the command-line interface. The **rpm** command is quite valuable and is worth understanding in some detail.

Take one of the RPM packages that you might need, **samba-swat**. For Red Hat Linux 9, it's also on the third installation CD. If you've followed the instructions in the last section, that CD should already be mounted.

It's easy to install the **samba-swat** RPM. The following **rpm** command uses the **-i** (for install) switch to install the RPM from the specified directory:

```
# rpm -i /mnt/cdrom/RedHat/RPMS/samba-swat-*
```

The asterisk is a wildcard; any RPM package file that starts with **samba-swat-*** is automatically installed. When you run the **rpm -i** command, there are three possible results:

- **rpm** successfully installs the package.
- **rpm** can't install packages because there are dependencies; in other words, you need to install other RPM packages first. If you've installed the **rpmdb-redhat** package as described in the previous section, the messages will name the packages that should be installed first.
- **rpm** can't install the package because it's already installed.

You could also find errors, which is the reason that more experienced Linux users add the **-v** (verbose) and **-h** (hash) switches to the command. Naturally, the **-v** switch provides additional information, and the **-h** switch creates hash marks, which can help you measure the progress of the installation:

```
# rpm -ivh /mnt/cdrom/RedHat/RPMS/samba-swat-*
```

But if an older version of this package is installed, this command won't update it. For that purpose, you'll want to replace the **-i** with a **-U** switch. The **-U** switch upgrades a package if it's already installed, or installs the package if it isn't currently on your system. So the preferred command is:

```
# rpm -Uvh /mnt/cdrom/RedHat/RPMS/samba-swat-*
```

The **rpm** command is rich and complex. Entire books are available on the subject. The first edition of a book named *Maximum RPM* is included on the Red Hat Linux 9 Documents CD. A few of the switches are shown in **Table 2**; they use the **samba-swat** package as an

example. Substitute the package of your choice. If the package is not located in the current directory, make sure to include the full directory name, such as /mnt/cdrom/RedHat/RPMS.

***Table 2**. Some common **rpm** command switches.*

Switch	Usage
-e	To delete a package such as **samba-swat**, use the **rpm -e samba-swat** command. Don't add the version number to the end of the package name.
-h	When you add the **-h** switch for installation, hash marks help you track the progress of the installation. This switch must be used with an installation switch such as **-i** or **-U**.
-i	To install a package such as **samba-swat**, use the **rpm -i samba-swat-*** command, where * is a wildcard that represents the version number of the package.
-q	To check if a package such as **samba-swat** is installed, use the **rpm -q samba-swat** command. Don't add the version number to the end of the package name.
-qi	If you want a full description of a package such as **samba-swat**, use the **rpm -qi samba-swat** command.
-ql	If you want a list of files included in a package such as **samba-swat**, use the **rpm -ql samba-swat** command.
--redhatprovides	If you want to identify the RPM package associated with a file such as /etc/passwd, run the **rpm --redhatprovides /etc/passwd** command. Note the double-dash in front of the switch.
-U	If you're not sure if there's an older version of **samba-swat** on your system, the **rpm -U samba-swat-*** command upgrades any older package, or installs the cited package if not already installed. Be careful; this switch uses a capital **U**.
-v	If you want more information from the output of a switch, add a **-v**. It does not stand alone.

This is just a short list of the switches that you can use with the **rpm** command. A more complete list is available in the **rpm** manual, which you can open at the command line with the following command:

```
# man rpm
```



*You can learn a lot about command-line commands by reading their manuals. For example, if you want to learn about how files are listed, run the **man ls** command.*

Uninstalling what's not necessary

If you have a large hard drive, you may not think too much about installing everything from the Red Hat Linux 9 installation CDs. Compared to the larger hard drives available, 5GB of software may not seem all that demanding.

However, extra software means extra vulnerabilities. You might not use every service that you have installed. Unused services are almost open doors for a cracker looking for a path into your system. There are two ways you can approach this issue: You can uninstall what you don't need, or you can deactivate a service that's installed. But first, let's take a look at the services that might be installed.

Checking installed services

In Red Hat Linux, network services are associated with two different directories. Major services are activated with scripts in the /etc/rc.d/init.d directory. Other network services can be found in the /etc/xinetd.d directory. From the command line, you can list the files in these directories. **Figure 11** illustrates a system with far too many services installed.

```
[root@localhost root]# \ls /etc/rc.d/init.d/
anacron  firstboot  irda       named    ntpd        saslauthd  sshd
apmd     functions  isdn       netfs    pcmcia      sendmail   syslog
atd      gpm        kdcrotate  network  portmap     single     winbind
autofs   halt       keytable   nfs      random      smb        xfs
crond    httpd      killall    nfslock  rawdevices  snmpd      xinetd
cups     iptables   kudzu      nscd     rhnsd       snmptrapd  ypbind
[root@localhost root]# \ls /etc/xinetd.d/
chargen      cups-lpd  daytime-udp  echo-udp  servers   sgi_fam  time-udp
chargen-udp  daytime   echo         rsync     services  time
[root@localhost root]#
```

***Figure 11**. Too many services on a Red Hat Linux computer.*

Believe it or not, this list is the result of the installation selections I made earlier in this chapter. For example, even though I did not select the Web Server package group, the Apache script, **httpd**, was still installed.



*Red Hat often installs "extra" software. Earlier in this chapter, I chose to install Red Hat Linux with the default Server Configuration Tools package group. This group includes **redhat-config-httpd**, which is a GUI tool that configures Apache. Thus, Red Hat assumed I forgot to include the Apache Web server in the installation list and "conveniently" added that software to the installed operating system.*

Before you make a judgment on whether to uninstall or deactivate a service, you might want to understand the services. I've listed the services from Figure 11 in **Table 3**. This is not a comprehensive list; it's based on the typical installation described earlier in this chapter. Be aware that only some of the scripts and services in the noted directories are network related. Don't uninstall or deactivate any of the non-network-related services unless you know what you're doing.

***Table 3**. Service scripts in /etc/rc.d/init.d.*

Script	Service	RPM package
anacron	Configures regular, scheduled, administrative jobs. Keep this package installed.	anacron
apmd	Controls Advanced Power Management (APM) features; Linux does not currently support the Microsoft-developed Advanced Configuration and Power Interface (ACPI). You should keep this installed.	apmd
atd	Supports the one-time scheduling of a specific job, such as processing a large database in the middle of the night. You should keep this installed.	at
autofs	Allows Linux to automatically mount partitions, CDs, and more. Must be installed.	autofs

Script	Service	RPM package
crond	Closely associated with anacron. Keep this package installed.	vixie-cron
cups	Supports the default Red Hat Linux print service of the same name. Don't uninstall unless you don't need to print anything locally or remotely from this computer.	cups
firstboot	Starts the First Boot utility described earlier in this chapter.	firstboot
functions	Contains the basic Linux boot scripts. Must be installed.	initscripts
gpm	Provides mouse support at text-based interfaces. Expected by most Linux users.	gpm
halt	Stops all Linux processes. Keep this installed.	initscripts
httpd	Controls the Apache Web server. Uninstall it if you don't need it.	httpd
iptables	Controls the default firewall. Uninstall it if you don't need it.	iptables
irda	Supports infrared connections. Uninstall it if you don't need it.	irda-utils
isdn	Supports connections through ISDN interfaces. Uninstall it if you don't need it.	isdn4k-utils
kdcrotate	Provides library support for for Kerberos 5 authentication. Uninstall it if you don't need it.	krb5-libs
keytable	Required for keyboards. Don't uninstall.	kbd
killall	Stops a group of processes associated with a specific name. Don't uninstall.	initscripts
kudzu	Starts the Red Hat hardware detection utility. Don't uninstall.	kudzu
named	Controls the Linux DNS server. Uninstall it if you don't need it.	bind
netfs	Mounts network filesystems. Don't uninstall.	initscripts
network	Controls network configuration. Keep this installed.	initscripts
nfs	Supports sharing between Linux and Unix computers using NFS. Uninstall if you don't plan to use NFS.	nfs-utils
nfslock	Supports locks on files on shared NFS directories. Uninstall it if you don't plan to use NFS.	nfs-utils
nscd	Caches hostname lookups.	ncsd
ntpd	Synchronizes the computer clock with a central time server. It's important to synchronize the time on your network.	ntp
pcmcia	Supports PCMCIA/PC Card hardware most commonly associated with notebook computers.	kernel-pcmcia-cs
portmap	Supports more secure network connections.	portmap
random	Controls random number generation. Used by encryption schemes and more. Don't uninstall.	initscripts
rawdevices	Required by certain applications.	initscripts
rhnsd	Allows connections by the Red Hat Update Agent to the Red Hat Network.	up2date
saslauthd	Supports authentication using the Simple Authentication and Security Layer (SASL).	cyrus-sasl
sendmail	Controls a popular e-mail server. Uninstall it if you don't plan to use an e-mail server on your computer.	sendmail
single	Moves Linux into single-user mode, which is similar to Microsoft Windows' safe mode without networking. Do not uninstall.	initscripts
smb	Controls Samba, which allows Linux computers to communicate on a Microsoft Windows-style network. Key for this book. Keep this installed.	samba

Script	Service	RPM package
snmpd	Controls utilities related to the Simple Network Management Protocol (SNMP), important for diagnosing network problems. Do not uninstall.	net-snmp
snmptrapd	Supports SNMP. Do not uninstall.	net-snmp
sshd	Controls the Secure Shell (SSH) service, which supports encrypted connections to remote computers. This is an excellent tool, which you should keep.	openssh-server
syslog	Controls utilities that support logging. Log information in /var/log is essential for the Linux system administrator. Do not uninstall.	sysklogd
winbind	Supports access to a Microsoft Windows database of user names and passwords. Important for this book. Keep this installed.	samba-common
xfs	Controls the X Font Server, closely associated with the X Window package group.	Xfree86-xfs
xinetd	Configures the extended Internet services daemon, which controls a number of other network services described later.	xinetd
ypbind	Allows connections as an NIS client. Uninstall it if you don't plan to use this method of creating a common password database.	ypbind

If you want to uninstall one of the services listed in Table 3, make a note of the name of the associated RPM package. You'll see in a moment how to use the **rpm** command and RPM package name to uninstall Apache.

As shown in Figure 11, there are a number of services associated with **xinetd**, in the /etc/xinetd.d directory. The scripts shown in that directory are fairly basic; you need not uninstall any of the associated packages. If you see additional services in that directory, read the instructions in the next section, which can help you determine whether you need the service.

Getting more information

If you've installed everything from the Red Hat Linux 9 installation CDs, you'll end up with twice the number of scripts in the /etc/rc.d/init.d directory. You should learn how to get more information on each script, so you can make your own decision about whether to uninstall or deactivate the package. Just follow these steps.

1. Read the script. Open the script of your choice in a text editor. You can use the text editor described earlier on the GNOME desktop by clicking Main Menu | Accessories | Text Editor. Alternatively, you can use the vi editor from the command line as described in Chapter 8, "Administration and Management." There are usually comments at the start of the file that explain its purpose.

2. Find the RPM package associated with the script. Assuming you've installed the **rpmdb-redhat** RPM discussed earlier in this chapter, it's easy to find the package name. For example, to find the package associated with the Apache (**httpd**) script, run the following command. Notice how I've included the full directory path to the file:

   ```
   # rpm --redhatprovides /etc/rc.d/init.d/httpd
   ```

 If there is a parent RPM package, you'll see it in the output. It includes the name of the package and the version number. If you've just installed Red Hat Linux 9, you'll see:

   ```
   httpd-2.0.40-21
   ```

3. Every RPM is supposed to include a description. You can find the description for the **httpd** package by using the following command. Note how I run this command *without* the revision number:

```
# rpm -qi httpd
```

Now that you know how to determine what's installed, you can make your own decisions about whether to uninstall or deactivate a service. I've noted several services in Table 3 that can be safely uninstalled or deactivated.

Uninstalling a service

If you know that you're never going to set up a Web server on the local computer, you can uninstall the associated packages. The same is true for the other services with scripts in the aforementioned directories.

It's easy to uninstall a package. For example, if you want to uninstall Apache, just run the **rpm** command with the name of the package (**httpd**):

```
# rpm -e httpd
```

But there can be problems. If other packages depend on what you're trying to uninstall, the attempt fails. But you should get a message about dependencies. An example of this is shown in **Figure 12**.

```
[root@localhost root]# rpm -e httpd
error: Failed dependencies:
        httpd is needed by (installed) redhat-config-httpd-1.0.1-18
[root@localhost root]# rpm -e httpd redhat-config-httpd
[root@localhost root]#
```

***Figure 12**. Uninstalling Apache and a dependency.*

There is a simple solution. As shown in Figure 12, you can use the same command to uninstall both packages simultaneously. For the configuration shown in Figure 11, I've repeated this command pattern to uninstall the DNS (**bind**), **sendmail**, **irda-utils**, and **isdn4k-utils** RPM packages. You can repeat this command for the RPM packages of your choice.

Deactivating a service

If you're planning to use a service in the near future, you can deactivate it instead of uninstalling it. Crackers can't get in through a deactivated service. Because this is a book for Microsoft Windows administrators who are converting to Linux, I start with the GUI tool, in this case, the Service Configuration utility. Start it by clicking Main Menu | System Settings | Server Settings | Services, which opens the Service Configuration window shown in **Figure 13**.

In the case shown in the figure, Apache is running and is set to start the next time you boot your computer. If you want to deactivate Apache, you need to take the following steps:

1. Highlight the service in question and then click Stop.
2. Deselect the service. This ensures that Linux does not start the service the next time you boot.

3. Repeat the process in the other runlevels. The key Red Hat Linux runlevels are 3 and 5. To switch runlevels, click the Edit Runlevel command in the toolbar, and then select the desired runlevel.
4. If you make changes, be sure to save them.



Red Hat Linux includes seven different runlevels. The important ones for our purposes are 3 and 5. In runlevel 3, you can log in to a text console. In runlevel 5, you can log in to a graphical console as shown in Figure 5. You can configure different services to start at each runlevel.

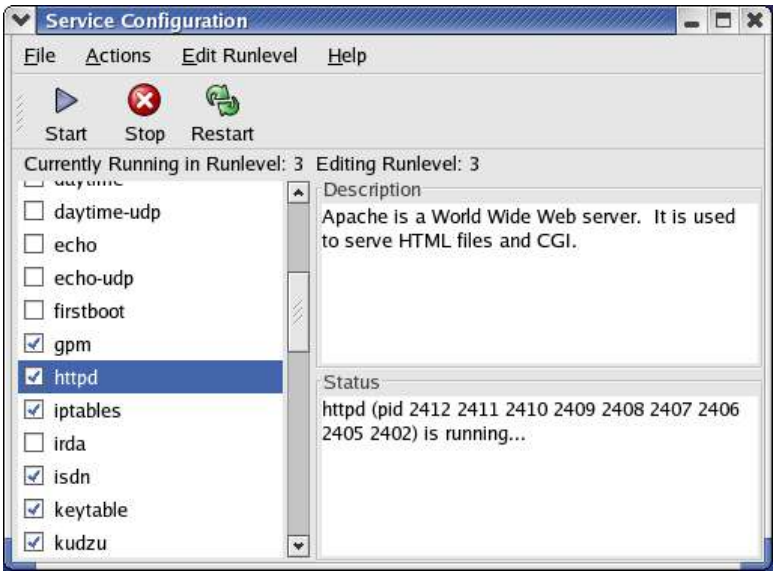


***Figure 13**. Checking Apache status in the Service Configuration window.*

You can also deactivate services at the command-line interface. For example, to deactivate the Apache (**httpd**) service at runlevels 3 and 5, run the following command:

```
# /sbin/chkconfig --level 35 httpd off
```

The **chkconfig** command is versatile; for example, you can read the list of active runlevels for **httpd** with the following command, which tells you if the service is set to be on or off at each of the seven different runlevels:

```
# /sbin/chkconfig --list httpd
```

Needless to say, it's almost as easy to make sure that a service becomes active when we boot to one of our key runlevels:

```
# /sbin/chkconfig --level 35 httpd on
```

Conclusion

This chapter started as a continuation of Chapter 1, finishing the installation process for Red Hat Linux. You can configure firewalls and authentication during installation. Most important is what you choose to install with Linux. If you install services that you don't need, you're essentially opening doors for crackers who want to break into your computer.

To this end, I examined the basic services that you need to configure a Linux File and Print server that can connect to a Microsoft Windows-style network. While the requirements are quite minimal, I've also specified the package groups that someone who is used to a GUI would want on Linux.

The First Boot process allows you to add a non-administrative user, synchronize your computer's clock with a central server, configure a sound card, set up a connection to the Red Hat Network (RHN), and install additional Red Hat software.

The Red Hat Update Agent allows you to update your software based on the latest available RPM packages on the RHN. While it requires registration, you can choose from any available updates.

After Red Hat Linux is installed, you may want to uninstall or deactivate several services with scripts in the /etc/rc.d/init.d directory. You can uninstall RPM packages by using the **rpm -e** command. You can deactivate services in the key runlevels (3, 5) by using the Service Configuration utility or the **chkconfig** command.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 3
Setting Up Your Server File System

In Chapter 2, I showed you how to install Samba on your Linux computer. Using Samba, you can make Linux look like a computer on a Microsoft Windows-based network. In this chapter, you'll learn how to set up Samba on a Linux computer. I'll show you what you need to configure to make a Linux computer with Samba work as a member server or a Primary Domain Controller (PDC) on a Microsoft Windows-based network.

Linux and Unix computers can share files and printers with each other. Microsoft Windows computers can share files and printers with each other. Unfortunately, the network file systems native to each operating system type are not compatible. That is the essential reason behind Samba, which allows Linux computers to understand the language of the Microsoft Windows network file system.

This chapter focuses on the details of configuring a Linux system as a File and Print server on a Microsoft Windows network. Because the file systems are different, I start this chapter by taking a look at the fundamentals of networking between Microsoft Windows computers, to define the different concepts of Workgroups and Domains.

Alternatively, you can set up the true 32-bit Microsoft Windows operating systems on a network of Linux and Unix computers with Microsoft Services for Unix. Unfortunately, this Microsoft service does not work with Windows 9x/ME computers. This chapter also covers some Linux fundamentals. I'll explain the basic services associated with Samba, along with the way that users and groups are organized in Linux.

Samba includes a number of configuration files. You'll learn about each of these files, as well as three ways to configure them. The easiest way to configure Samba is with Red Hat's Samba Server Configuration tool, which I colloquially refer to as the Samba Server Configurator. Unfortunately, it doesn't allow you to take full advantage of what Samba can do on a Microsoft Windows network. The Samba Web Administration Tool (SWAT) allows you to configure every Samba setting. While it's a GUI browser-based tool with links to all sorts of help, it is long and complex. Finally, you can edit the main Samba configuration file, smb.conf, directly in the text editor of your choice.

Finally, I cover the details of how Samba really works through its main configuration file, smb.conf. The default version of this file includes a number of clues and suggestions for how to make your Linux computer look like a Microsoft Windows server on a Workgroup or a Domain. With Samba, you can also turn your Linux computer into a Windows Internet Name Service (WINS) server and a Domain Master Browser. I'll show you how to edit smb.conf directly to configure shared directories and printers.

Most of the commands in this section require the use of the Linux administrative account, root. If you've logged in as a regular user (as you should), you can get root user access by using the following commands:

```
$ su
Password:
#
```

The Microsoft CIFS

In short, the Microsoft Windows Common Internet File System (CIFS) describes how Microsoft Windows networks communicate on TCP/IP networks. While it's all based on the Server Message Block (SMB) protocol, Microsoft uses CIFS to emphasize the interconnectivity of its networks with the Internet.

There is a whole history behind networking on Microsoft Windows computers. While the following descriptions are "oversimplifications," they give you a feel for the complexity of Windows networking. However, the GUIs for Windows networks share a common look and feel. Networks can be organized into small groups of computers known as Workgroups, and larger groups with centralized systems known as Domains.

Before we continue, make sure you understand both the Workgroup and Domain concepts as they apply to Microsoft Windows-based networks.

Workgroup: A simple group of computers connected together in a network. Any computer in a Workgroup can share directories and printers. A Workgroup does not have a centralized database of users and passwords. This is a common way to organize networks of 10 or fewer computers, and is also known as a "peer-to-peer" network.

Domain: In the Microsoft world, this is a group of computers, connected together in a network, with a centralized database of users and passwords. This type of network often is administered with common policies for users and computers. This is different from domain names on the Internet; those names relate to a hierarchy of computers on that worldwide network.

Background

A network consists of two or more computers that communicate with each other. Generally, networks are used to share files and printers. When Microsoft first wanted to set up File and Print servers, it borrowed from its operating-system roots with IBM and adapted NetBIOS (the Network Basic Input Output System). Computer names are limited to 15 characters. Microsoft also adapted IBM's enhanced NetBIOS system, NetBEUI (NetBIOS Extended User Interface), which supports individual networks of up to 255 computers.

The weakness of NetBEUI is that it is not routable; in other words, you can't set up NetBEUI to allow two different networks to communicate with each other. The development of the Internet stimulated Microsoft to take its network interfaces in a different direction, which led to the development of CIFS. Because it is still based on SMB, CIFS has retained its compatibility with Samba.

In modern networks, there are clients and servers. Servers offer to share directories and printers with other computers on the network. Clients connect to and use these "shares."

My Network

In this book, I've set up a number of different computers with various Microsoft Windows operating systems, from Windows 95 through Windows XP Professional. I've also set up three computers with Red Hat Linux 9 in various roles. I've set them up in a peer-to-peer Workgroup named Darkstar and a Domain named Grateful. Each of the individual computers listed in **Table 1** is named after various Grateful Dead Bean Bears.

***Table 1**. Computers on my network.*

Computer Name	Operating System	Role
sugaree	Windows NT 4 Server	PDC
cosmicc	Red Hat Linux 9	PDC
tennjed	Windows 2000 Server	Domain Member Server
allaccess	Windows XP Professional	Client
reuben	Windows 98	Client
ststephen	Windows 2000 Professional	Client
nopaws	Red Hat Linux 9	Domain Member Server
samson	Windows 98	Client
daisy	Windows NT Workstation	Client
ripple	Windows 95	Client
althea	Windows ME	Client
delilah	Red Hat Linux 9	Client

The clients in this network can be part of the Darkstar peer-to-peer Workgroup or the Grateful Domain. I can configure them to share local directories or printers with other members of the Workgroup or Domain. Several of these computers are installed as "Virtual Machines" using VMWare (**www.vmware.com**). While I have a Windows and a Linux computer configured as a PDC, I do not run both computers at the same time.

Basic look and feel

On the various Microsoft Windows operating systems, there are three basic ways to view computers and shared directories on a network. Older Microsoft operating systems, namely Windows 95/98, support a viewer known as the Network Neighborhood. The example shown in **Figure 1** is from a Windows 98 computer connected to a Workgroup named Darkstar and a Domain named Grateful.

You'll note that Workgroups and Domains look alike in the GUI. Both are groups of computers governed by common sets of rules. But as you already know as a Microsoft Windows administrator, Domain user names, passwords, and profiles are stored in a centralized location.

Newer Microsoft Windows operating systems, including Windows ME, work with a slightly different GUI view of networks, known as My Network Places. As you can see in **Figure 2**, this is a broader view of networking which, interestingly enough, can include the NFS (Network File System) networks that are native to Linux and Unix computers. I'll discuss that option in more detail later in this chapter.

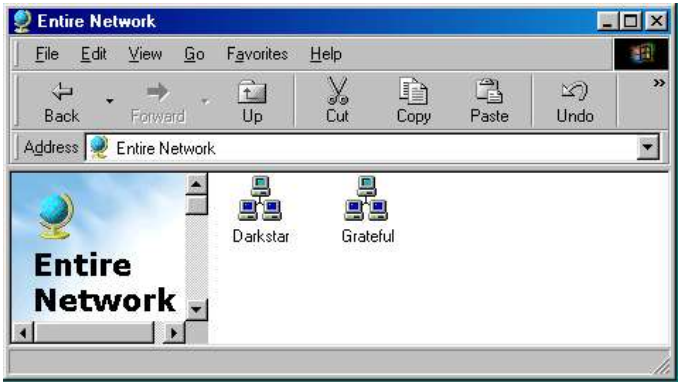


***Figure 1**. The Microsoft Windows Network Neighborhood, showing the Darkstar Workgroup and the Grateful Domain.*

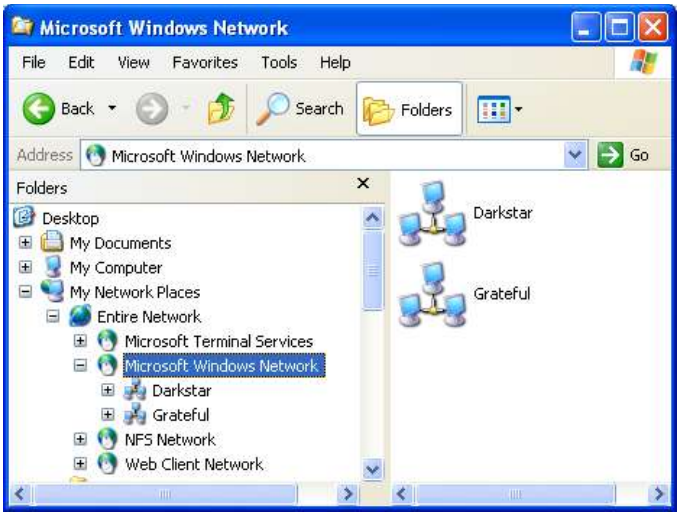


***Figure 2**. My Network Places.*

Although the standard way to view networks in Microsoft Windows is based on GUI interfaces, you can trace Microsoft's network utilities back to text commands that you can run at the Microsoft command prompt.

In any version of Microsoft Windows, open a command-line interface. Click Start | Run and type **cmd** in the text box. In the command-line window, try the following command:

```
C:\> net view
```

If your computer is connected to a Microsoft Windows network, you'll see the computers in your Workgroup or Domain, depending on your permissions. On a computer in my personal Windows Workgroup, that leads to the following output:

```
Server Name            Remark
_____________________________________________

\\ALLACCESS            LaptopWin
\\COSMICC              Samba Server
\\TENNJED              w2000svr
The command completed successfully.
```

This output includes the names of computers on my Darkstar Workgroup. If you run the same command on a computer on a Domain, you'll get a list of the computers that are connected to that Domain. For example, the following DOS command lists the computers in my Grateful Domain:

```
C:\> net view /Domain:grateful
```

When you configure a Linux computer with Samba, you can choose whether it's a client or a server—and on either a Microsoft Workgroup or a Domain.

Before I show you how to configure Linux computers on a Microsoft Windows network, I'll show you an alternative: configuring NFS on a Microsoft Windows computer.

The NFS alternative—an overview

The Network File System (NFS) is designed for sharing directories between Linux and Unix computers. NFS is native to Linux, so if you have only Linux and Unix computers on your LAN, it's more efficient to use NFS. You could set up a Samba network between Linux and Unix computers, but why install software that you don't really need? There are security issues associated with NFS, but you can protect an NFS LAN with an appropriate firewall.

But the point of this book is to set up communication on a network with Linux and Microsoft Windows computers, which is why I focus on Samba.

Microsoft does offer a way to communicate with NFS networks: Microsoft Services for Unix 3.0. For an additional $99 per computer, you can set up your Microsoft 32-bit Windows computers as clients and servers on an NFS network. However, this does not work for Microsoft Windows 9x without additional third-party software. It also does not work on the Microsoft Windows XP Home operating system.

Thus, unless you're willing to limit the Microsoft Windows clients on your network to Windows NT, 2000, and XP Professional, Microsoft Services for Unix is not a viable option. Therefore, I focus on Samba as a way to help Linux communicate on a network with Microsoft Windows computers.

Basic administration

When you administer a Linux computer, you need to understand processes and users. Services such as Samba run as processes. When you configure Samba, you'll generally need to configure a file to associate Microsoft Windows user names with Linux users.

As you'll see in Chapter 4, "Setting Up Your File Server's Users," you could set up the same user names on a Linux server, and then use Samba commands to make them readable on a Microsoft network. If you can't set up the same user names, you could add a database that matches the user names you have on your Linux Domain server and the Microsoft computers on your network.

Process management

You've presumably installed Samba as part of the Linux installation as described in Chapter 2, "Installing Linux as a File Server." Now you need to know how to manage the Samba processes —specifically, how to start and stop Samba, as well as reload the Samba configuration file. You can manage services from a GUI application or from the command-line interface.

Managing services from a GUI

To manage Samba in a GUI, start the Service Configuration utility first described in Chapter 2. Click Main Menu | System Settings | Server Settings | Services. You can scroll down the list of running services to review the status of the Samba and Winbind services, as shown in **Figure 3**.

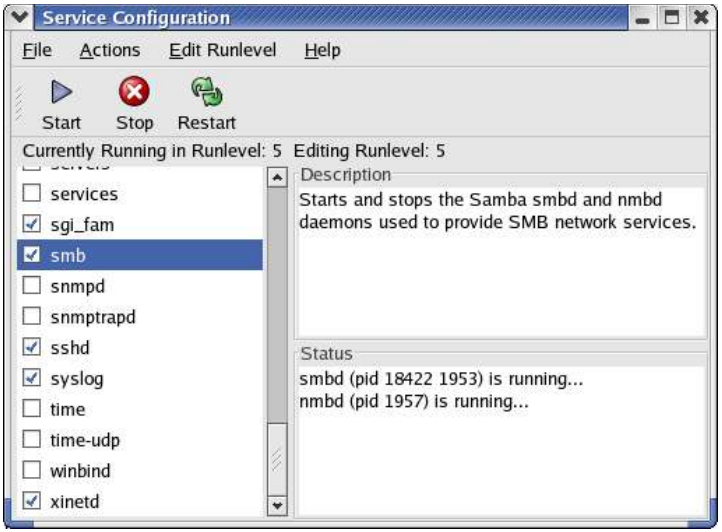


***Figure 3**. Checking the status of Samba.*

As you can see from the figure, the Service Configuration utility is checking service status in runlevel 5. As described in Chapter 2, that's where you log in to a GUI interface. I've highlighted **smb**; in the Status pane, you can see the status and Process IDs associated with the Samba (**smbd**) and NetBIOS (**nmbd**) daemons. These are the programs in the /usr/sbin directory that actually run Samba on Linux.



*Process IDs (PID) can help you keep a crashed program from crashing your system. While you can select a process and end it as shown in **Figure 4**, you can also run the **kill -HUP PID** command for a graceful shutdown (or run **kill -9 PID** for an immediate shutdown that may leave orphan processes running).*

You might also note that the **winbind** service is not active. If you want to set up Linux member servers to use a Microsoft Windows database of users and passwords, you'll want to activate this daemon with the **/sbin/service winbind start** command. I discuss this daemon in more detail in Chapter 4, "Setting Up Your File Server's Users."

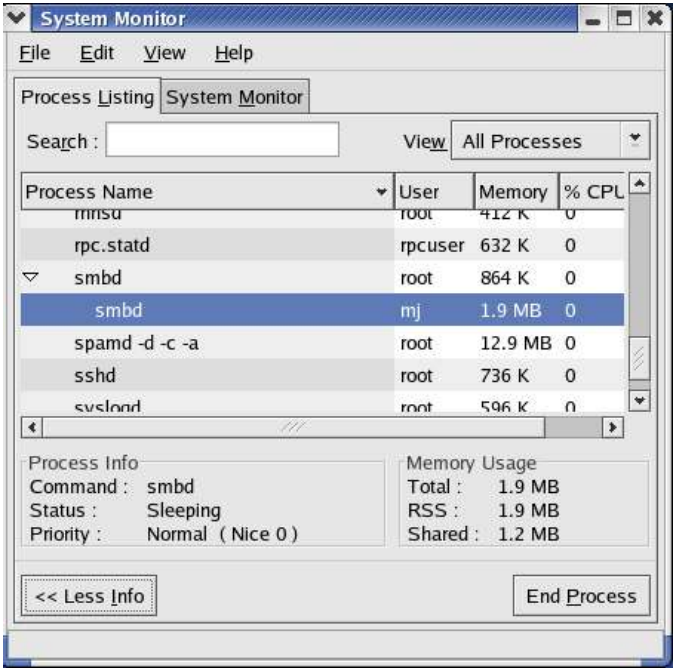


***Figure 4**. Monitoring system processes.*

You may also want to check the memory and CPU used by various services. You can do so in Red Hat Linux with the System Monitor, which is functionally similar to the Microsoft Windows Task Manager. To start the System Monitor, click Main Menu | System Tools | System Monitor. Take a look at this tool in Figure 4. It's easy to identify the amount of memory and Process IDs associated with the Samba daemon, **smbd**.

Note the two different **smbd** processes in the list. The basic Samba script is associated with the root user; you can also see that someone has connected to a shared Samba directory as the user **mj**.



*Seasoned Linux users believe it's important to know how to manage processes from the command-line interface. For example, if you have a crash that affects the GUI, you can still keep your server working without rebooting or disconnecting remote users. You can review your processes with the **ps** command. You can review all processes on your computer with the **ps aux** command. In addition, you can monitor currently running processes by using the **top** command.*

User categories

When you administer a server, part of your job is to administer user accounts. Like any Microsoft Windows server, Linux depends on user names and passwords. Linux classifies users

into slightly different categories, as you can see by reviewing the main user name/password authentication file, /etc/passwd. There are three categories of users:

- The administrative user on a Linux computer is **root**. Because this book is for administrators, I assume you have access to this user account.
- Services are associated with their own user names, such as **mail**, **news**, and **apache**. You can see a full list in your /etc/passwd file.
- Individual users have their own accounts.

In Red Hat Linux, each user gets an individual ID (UID), which by default starts with 500. Each user is assigned to his or her own group. For example, on my Red Hat Linux computers, I normally set up a regular account with a user name of **mj**. Red Hat Linux by default also assigns user **mj** to a group named **mj**. This is known as the User Private Group scheme, which is designed to keep individual users away from the files of others and is commonly used by Internet Service Providers (ISPs). As you'll see in Chapter 4, "Setting Up Your File Server's Users," you can use the Red Hat User Manager to set up users with names, home directories, account and password rules, and more.

You'll also see the details on how to make the Red Hat User Manager work with a Samba user database. But before you create Samba users for your Windows network, you need to know how to configure a Samba server.

Configuring a Samba server

You probably installed the necessary Samba RPM packages when you installed Red Hat Linux. You may be able to get more up-to-date versions of Samba from the Red Hat FTP site or from the Rawhide directories. For more information, see Chapter 2, "Installing Linux as a File Server." Once Samba is installed, you'll want to understand the key configuration files. If your Samba server is on a firewall, you need to make sure Samba messages can get through.

Then you can finally configure Samba itself by using one of two GUI tools: the Samba Server Configurator or the Samba Web Administration Tool (SWAT). Later in this chapter, I'll show you how to edit the Samba configuration file directly from the command-line interface.

Getting Samba

Here's the easiest way to see if you successfully installed the RPM packages that you need. Run the following command:

```
# rpm -qa | grep samba
```

If successful, you'll see the following packages in the output. You might see different version numbers; these packages are from the Red Hat Linux 9 installation CDs.

```
samba-client-2.2.7a-7.9.0
samba-2.2.7a-7.9.0
samba-swat-2.2.7a-7.9.0
samba-common-2.2.7a-7.9.0
redhat-config-samba-1.0.4-1
```

If you don't have all of these packages on your computer, you can install them from the installation CDs, from the Rawhide directory on a Red Hat FTP server, or through the Red Hat Update Agent. For example, if you want to install **samba-swat** from the Red Hat Linux 9 installation CDs, follow these steps as the root user:

1. Insert the first CD into the drive. If you're in the Linux GUI, Linux normally mounts the drive automatically, and then opens a Nautilus window with a view of the files and directories in that drive.
2. Open a command-line interface. If the CD wasn't mounted, run the following command:

   ```
   # mount /mnt/cdrom
   ```

3. To install the **samba-swat** RPM, run the following command:

   ```
   # rpm -Uvh /mnt/cdrom/RedHat/RPMS/samba-swat*
   ```

There is another source for the latest version of Samba; you can download it directly from **www.samba.org**. Unfortunately, the packages on this Web site are organized in source code, which you would have to compile, with appropriate options, to binary code before installation. This process might be difficult for those who are new to Linux; if you're interested, read the Unofficial Samba HOWTO at **hr.uoregon.edu/davidrl/samba/**. I prefer to install Samba from Red Hat RPMs, which are optimized for this Linux distribution.

Now that you've installed the appropriate packages, it's time to look at the key files.

Samba configuration files

There are five key configuration files associated with Samba, located in the /etc/samba directory. They define how your Linux computer interacts on a Microsoft Windows-based network.

- **lmhosts** includes a database of IP addresses and NetBIOS names. It's similar to the LMHOSTS file that you might find on a Microsoft Windows server. It's rarely used, given the available alternatives for finding computers: a Windows Internet Name Service (WINS) server, a DNS server, or even the hosts file in the /etc directory.
- **secrets.tdb** contains a Microsoft-style security identifier (SID) for your Samba computer. Because it's in binary code, you can't read it through a text editor.



*If you install or upgrade to Samba version 3.0.x, you can find your SID by using the **net getlocalsid** command. As of this writing, this is available by download from the Rawhide directory on **ftp.redhat.com**.*

- **smb.conf** is the main Samba configuration file. You'll configure this file in detail using GUI tools, and then analyze this file later in this chapter.
- **smbpasswd** contains the encrypted password associated with each Microsoft Windows-style user name, as documented in smbusers. The password in this file can

be different from the password for the corresponding Linux account. Those who connect from Microsoft Windows workstations (including Samba clients on Linux computers) need this password.

- **smbusers** includes a database of users on the local computer and the corresponding Microsoft Windows user name. It also includes the computer machine accounts of Microsoft computers that may connect to this network.



As you'll see in Chapter 4, "Setting Up Your File Server's Users," you can also set up a Domain member server on Samba that works with a Linux or Microsoft PDC.

Samba and firewalls

This section shows how to create a firewall on a Linux file server. The standard Red Hat Linux firewalls don't allow communication with a Microsoft Windows network. I'll show you how to customize standard-, medium-, or high-security firewalls to support such communication.

In most cases, you'll install Samba on a Linux computer on a network that's already protected by a firewall. You don't need another firewall on the local computer. You can deactivate any local firewalls by deactivating the iptables script through the Service Configuration utility—or by deleting the **iptables** RPM, as described in Chapter 2, "Installing Linux as a File Server."



Don't allow access to Samba to an insecure network such as the Internet. Samba uses the same Internet ports as Microsoft Windows, which are known to be vulnerable to attacks over the Internet.

If you're responsible for several networks, you might be protecting them from each other with a firewall. You can configure Samba services through those internal firewalls. Essentially, you need to make sure that five of the 65,536 TCP/IP ports are open to allow Windows and Linux clients to communicate with your Samba server through the firewall.

While I could get into a complex discussion of the rules associated with **iptables**, the easiest way to create this type of firewall is with the lokkit utility. Open a command-line interface, and start it as the root user with the **/usr/sbin/lokkit** command. This opens the lokkit screen shown in **Figure 5**, where you can create a high-security or medium-security firewall. Assuming you actually need a firewall on this particular computer, select the level that most closely reflects the security you need on your network.



If your Linux computer already includes a firewall, don't use lokkit. It erases any rules that you may have previously configured, such as the rule described in Chapter 2 to synchronize your computer to a remote time server. Instead, you'll need to incorporate the rules listed at the end of this section.

Once you've selected the firewall, press the Tab key to highlight the Customize option, and then press Enter. This opens the Customize window shown in **Figure 6**, where you can allow incoming traffic at least for your Samba server. The available options relate to services as described during the installation process in Chapter 2, "Installing Linux as a File Server."

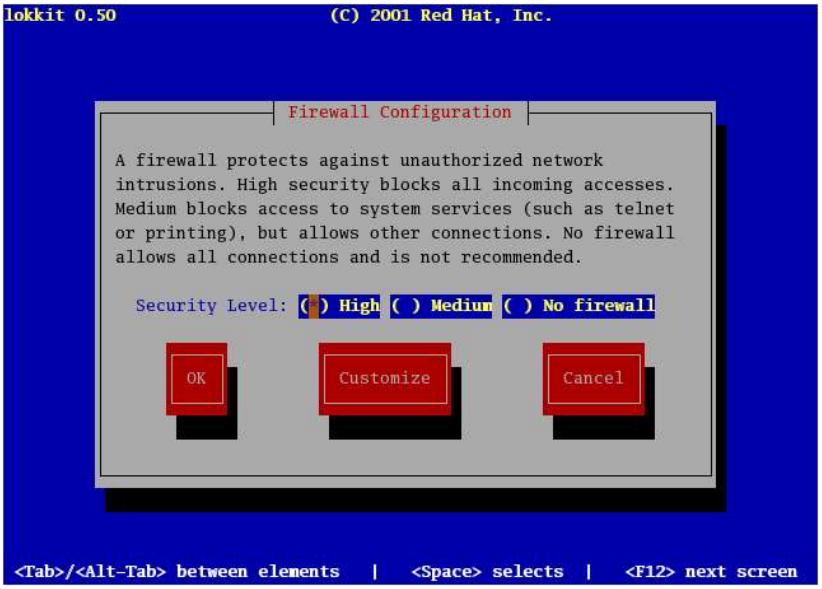


***Figure 5**. Using lokkit to configure a high-security or medium-security firewall.*

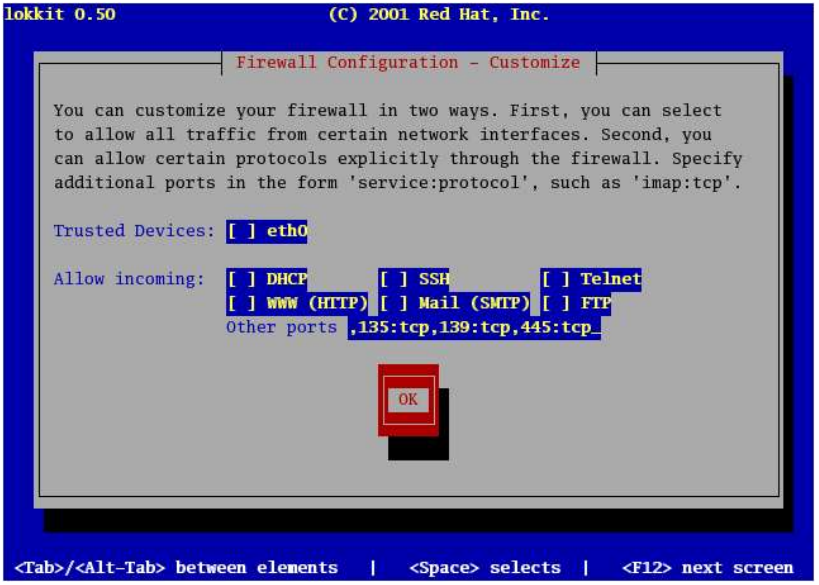


***Figure 6**. Using lokkit to customize a firewall.*

To support Samba traffic with Microsoft Windows clients, you need to enter the following line in the "Other ports" text box:

```
135:udp,137:udp,138:udp,1512:udp,135:tcp,139:tcp,445:tcp,1512:tcp
```

This allows lokkit to set up the firewall rules that you need to support Samba communication with clients outside your network. For more information on each of these port numbers, see **Table 2**.

***Table 2**. Samba TCP/IP ports for configuring a firewall.*

Port	Function
135	Remote Procedure Call (RPC) client mapper. Supports communication from clients.
137	NetBIOS name service. Allows transmission of NetBIOS computer names.
138	NetBIOS datagram service. For communication that doesn't require a response.
139	NetBIOS session service. Supports reliable data transmission.
445	Supports Samba connections with Windows 2000/XP clients.
1512	Allows communication with a WINS server.

For those of you who know **iptables**, you can add these commands to your firewall configuration file to allow Samba communication through that firewall (In Red Hat Linux, the default firewall configuration file is /etc/sysconfig/iptables):

```
iptables -A RH-Lokkit-0-50-INPUT -p udp -m udp --dport 135   -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p udp -m udp --dport 137   -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p udp -m udp --dport 138   -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p udp -m udp --dport 1512  -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 135 --syn -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 139 --syn -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 445 --syn -j ACCEPT
iptables -A RH-Lokkit-0-50-INPUT -p tcp -m tcp --dport 1512 --syn -j ACCEPT
```

There is one more way to regulate access to Samba: by using the **hosts allow** and **hosts deny** variables in the smb.conf configuration file. The default version of this command restricts access to networks with the following IP network addresses: 192.168.1.0 and 192.168.2.0, as well as the loopback address, 127.0.0.1, to allow access from the local computer. Here is the default version of this command:

```
; hosts allow = 192.168.1. 192.168.2. 127.
```

We'll take another look at this command later. Let's start easy with the Red Hat GUI utility, which is designed to help you configure a Samba server on your computer.

The Red Hat Samba Configurator

Red Hat has developed a fairly simple way to configure a Samba server. While its configuration capabilities are limited, it's sufficient for many purposes. This tool is included with the **redhat-config-samba** RPM package. You can start the Samba Configurator from the GUI by clicking Main Menu | System Settings | Server Settings | Samba Server. See **Figure 7**.

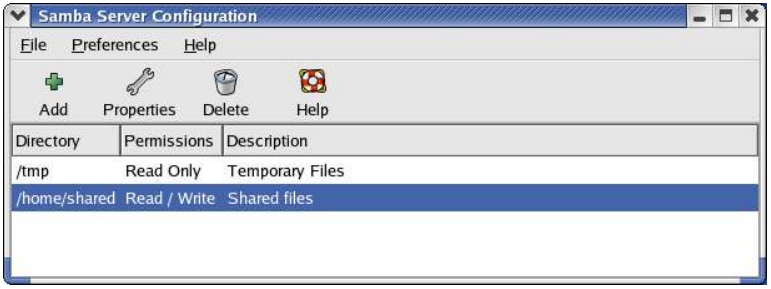


***Figure 7**. The Red Hat Samba Server Configurator.*

Before you begin, back up the standard Samba configuration file, /etc/samba/smb.conf. It includes some excellent comments that you can use to learn about a number of Samba variables. To configure a Samba server, it's important to start by learning about a few basic Samba variables:

- **Authentication Mode**: Corresponds to the Samba **security** variable. There are four options: **share**, **user**, **server**, and **domain**. As strange as it sounds, if you're setting up your Samba server as a Primary Domain Controller (PDC) on a Microsoft network, you'll want to set this mode to **user**.
 - **share**: Supports shared directories on Workgroups common to Windows 9x; shared directories are protected only by passwords.

 ! *Linux security is based on user names and passwords. If you set up share-level security, you could compromise your system to anyone who learns a critical password.*

 - **user**: Requires a user name and password for access to shared directories on the server or Domain. Checks authentication against local databases. Appropriate for a member server on a peer-to-peer Workgroup—or a Domain controller. For example, if you're configuring a computer as a PDC, you want remote users to use local user names and passwords as authoritative for the Domain.
 - **server**: Requires a user name and password for access to shared directories on the server or Domain. Checks authentication against a database on a remote computer. If the connection to the remote computer is down, this reverts to **security = user**. Normally you wouldn't use this option on a Domain.
 - **domain**: Requires a user name and password for access to shared directories on the server or Domain. Checks authentication against a database on a remote Domain controller. If the connection to the remote computer is down, this reverts to **security = server**. Appropriate for a Linux member server or workstation client on a Domain. Not appropriate for a PDC, because you want a PDC to have the final say on user names and passwords on a Domain.

- **Authentication Server**: If the authentication database is on a remote computer, it's called the authentication server, which corresponds to the Samba **password server** variable. In other words, if you're setting up a Samba member server on a Domain, you'd list the NetBIOS name of the PDC here.
- **Encrypt Passwords**: Passwords are and should be encrypted by default on Microsoft Windows networks. This can cause problems with computers running older versions of Microsoft Windows, such as Windows for Workgroups 3.11 or Windows 95 without Service Pack 2 (also known as OSR2, released in 1996). Corresponds to the Samba **encrypt passwords** variable. (If you are using these older Microsoft systems and are willing to accept passwords in clear text on your network, you can set this variable to **no**. I explain how to set up clear text passwords in Chapter 6, "Connecting Windows Workstations.")
- **Guest Account**: If you've configured share-level security, you'll want to set up a guest account as the default user. Before assigning a guest account, I recommend that you create a dedicated account for this purpose. For more information on creating accounts, see Chapter 4, "Setting Up Your File Server's Users."

Now that you understand the basics of Samba security, you can configure a Samba server. In the following example, I configure a Samba server for project access to a group of two users; their Windows user names are Nancy and Michael. Their Linux user names are **nc** and **mj**. They'll share access to the /home/project directory. To set this up, follow these steps:

1. In the Samba Server Configuration dialog, click Preferences | Server Settings to open the Server Settings window. On the Basic tab, you'll see the Workgroup and Description text boxes. In the Workgroup text box, enter the name of the Windows Workgroup or Domain that you want this Samba server to join. In the the Description text box, enter the comment of your choice. Then click the Security tab shown in **Figure 8**.



***Figure 8**. Configuring Samba security.*

2. Earlier in this section, I described the available security options. Make any desired changes on the Security tab and click OK to return to the Samba Server Configuration window.

3. Configure Samba users. You'll need a list of users on the Linux computer from /etc/passwd, and Microsoft Windows users from the Domain controller or individual workstations. When you have this information, click Preferences | Samba Users to open the Samba Users dialog.
4. Click Add User to open the Create New Samba User window shown in **Figure 9**.

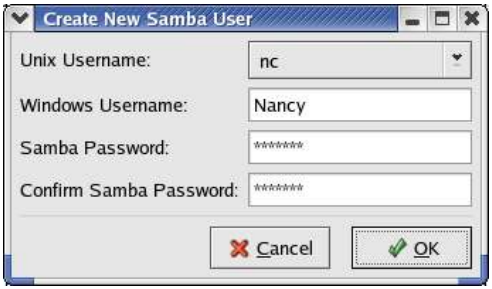


***Figure 9**. Setting up a database of Linux and Windows users.*



In most Linux documentation, the terms "Unix" and "Linux" are interchangeable. Remember, Linux is a clone of Unix, so they function in the same way.

5. You can map each Linux user on your computer to the Microsoft Windows user names on your network. The Linux user names must already exist. You can select such user names from the Unix Username drop-down text box. The Windows user names may be new or may already exist on an external computer. The Unix user name is what a user on a Samba or Windows workstation would use to log in to your Samba server or Samba-enabled Domain. These user names and passwords are added to the smbusers and smbpasswd files. Click OK when complete.



As of this writing, for Red Hat Linux 9, you'll also have to activate the following command in the /etc/samba/smb.conf file:

```
username map = /etc/samba/smbusers
```

6. Repeat steps 4 and 5 as desired to create the users that you need to log in from workstations on this Samba network. Then click OK to return to the Samba Server Configuration window.
7. Assume you're working with a consultant who needs a shared directory for the latest project. Click Add to open the Create Samba Share window shown in **Figure 10**.



***Figure 10**. Configuring a shared Samba directory.*

8. On the Basic tab, enter the directory you want to share; this corresponds to the Samba **path** variable. The description is a Samba **comment** variable that corresponds to the directory in Windows Network Neighborhood or My Network Places. You can set the directory with read-only (**read only = yes**) or read-write (**writable = yes**) permissions.
9. Click the Access tab. You can allow access to all users, or limit it to specific users from the smbusers file. For this particular project, I'm limiting access to the /home/project directory to the specific users **mj** and **nc** (see **Figure 11**). When you've made desired changes, click OK. The changes are documented in the Samba Server Configuration window.

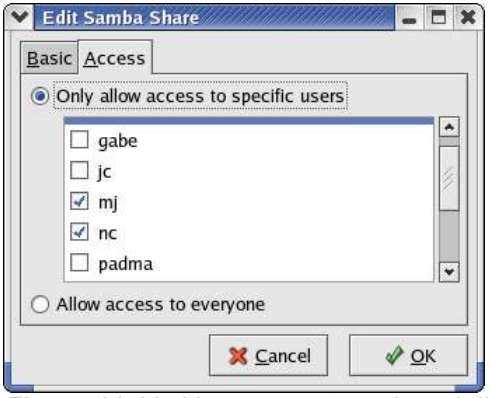


***Figure 11**. Limiting access to a shared directory.*

10. If you want to change the settings associated with any shared directory, highlight it and click Properties. When you're done with your changes, click File | Quit.
11. Finally, you need to let the Samba service reread the configuration file. You can stop and start **smb** in the Service Configuration utility. Alternatively, you could enter the following at the command-line interface:

```
# /etc/rc.d/init.d/smb restart
```

Now Michael and Nancy can both share files in the /home/project directory from their Linux or Windows workstations on the network. You can see how this works from a Linux or Windows workstation in Chapters 5 and 6. While the Samba Server Configurator is easy to use, there's an alternative that can help you configure Samba in almost excruciating detail: the Samba Web Administration Tool (SWAT).

Samba Web Administration Tool

SWAT allows you to configure Samba through a Web browser interface. To use it, you need the **samba-swat** RPM. After you install that, you can activate the **swat** service in the /etc/xinetd.d directory. To do so, run the following command:

```
#/sbin/chkconfig swat on
```

This command activates SWAT, and ensures that the **swat** service starts the next time you boot Linux. Start the browser of your choice. Depending on what you have installed, you can choose from a number of different Web browsers. Click Main Menu | Internet | More Internet Applications. You can start the standard Mozilla Web Browser, or several other browsers such as Balsa or Konqueror, assuming they're installed.



Before editing your Samba configuration, back up the default /etc/samba/smb.conf file. It includes a number of excellent comments, which SWAT overwrites. I will go through the default smb.conf file and explain some of these comments later in this chapter.

As long as you've activated **swat**, all you need to do is type **localhost:901** in the address box. You should be prompted for the root user name and password. You can review the initial SWAT menu in **Figure 12**.

The SWAT home page includes links to a considerable amount of documentation, including daemons and configuration files; utilities and tools that you can run from the command-line interface; and HOWTOs from **www.samba.org** that were current when your version of Samba was released.

There are a number of options associated with most SWAT menus, described in **Table 3**. Some of these options exist only in other SWAT submenus as described in the following sections.

You can find out more about each Samba variable in SWAT; just click the Help link to the left of the desired variable. If you want to find the Samba default for a variable, click the associated Set Default button. Just be careful; the results may not be desirable. You can restore the original settings when you click the Reset Values button. If you want to configure Samba variables that you don't see, click the Advanced View button. When you're done making changes, click Commit Changes.

Now I'll take a brief look at the other menus, which are accessible through the links at the top of the Web page.



***Figure 12**. The main SWAT menu.*

***Table 3**. SWAT menu options.*

Option	Function
Home	Returns to the main SWAT menu, which includes links to documentation.
Globals	Allows you to configure basic Samba server parameters, including Workgroup/Domain membership and security.
Shares	Permits you to configure shared directories.
Printers	Lets you share installed printers.
Wizard	Allows you to configure a basic Samba server.
Status	Provides a screen to review the status of Samba daemons and any mounted directories.
View	Allows you to review the smb.conf file.
Password	Supports an interface to manage local and remote user names and passwords.

SWAT Globals menu

Click Globals on the SWAT home page. As shown in **Figure 13**, this screen allows you to configure the basic parameters for the Samba server. It includes the options you saw in the Server Settings window of the Samba Configuration tool (Figure 8), and a lot more.

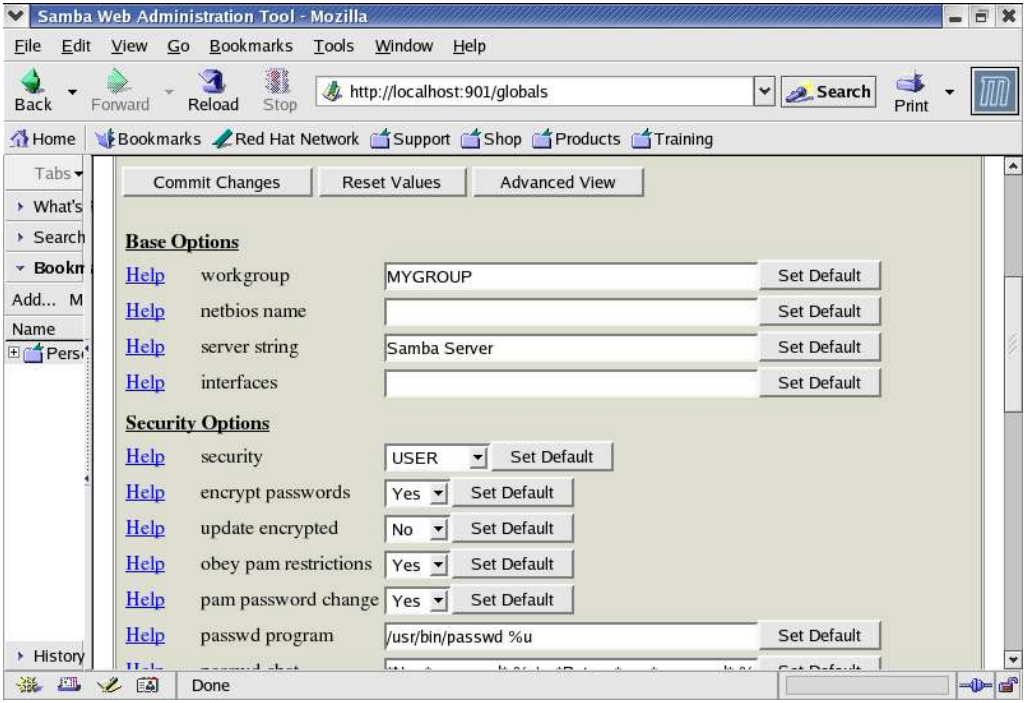


***Figure 13**.SWAT Globals menu.*

The two defaults that you're most likely to change are **workgroup** and **security**. The **workgroup** can be the name of your Microsoft Windows Workgroup or a Domain. Security, as described earlier in the Samba Server Configurator, can be **share**, **user**, **server**, or **domain**. You should also set up a NetBIOS name to make your computer visible on Microsoft Windows networks.

There are several other options of particular interest that are beyond the capabilities of the Samba Server Configurator:

- **Security Options**: While the password parameters rarely change, you can restrict access to specific networks with the **hosts allow** and **hosts deny** variables. For example, if you want to limit access to computers with the 172.22.30.0 network (and the local computer), enter the following value in the **hosts allow** text box:

```
172.22.30. 127.
```

- **Browse Options**: Microsoft Windows networks maintain a browse list of computer NetBIOS names. You can set up your Samba computer as a browse master. If your computer is the PDC and you absolutely want your Samba server to be the browse master for the Domain, set the following parameters:

```
os level = 255
preferred master = yes
local master = yes
domain master = yes
```



*When you configure Samba, the default **os level = 20** is normally sufficient unless you're configuring this computer as the PDC. I describe the standard **os level** options later in this chapter.*

- **WINS Options**: The Windows Internet Naming Service (WINS) collects a database of computer names and IP addresses on a Microsoft Windows network. If you set a **dns proxy**, Samba can search for computers on your network with a DNS server as well. If you have a remote **wins server** on your network, specify its IP address and fully qualified domain name (FQDN). Alternatively, you can set the local computer as the WINS server with the following parameter:

```
wins support = yes
```



Don't set up more than one Linux-based WINS server on your LAN. You may want to set up other WINS servers on neighboring LANs on the same Domain.

SWAT Shares menu

Now to configure a shared directory, click Shares. Select an existing share from the Choose Share drop-down list and then click the Choose Share button. Alternatively, you can create a new share by typing a name and clicking Create Share. The basic Web interface is shown in **Figure 14**. The options are more extensive than what you saw in the Create Samba Share window of the Samba Server Configurator.

Naturally, you'll want to specify the path to the directory that you're sharing, as well as a comment to help users understand the content of the directory. While by default, shared Samba directories are **read only = yes**, you can make a shared directory writable for authorized users by setting **read only = no**.

You can set security options for each individual share. For example, you can limit access to specific computers or networks with the **hosts allow** and **hosts deny** variables. There are two other basic variables of interest: **browseable** and **available**. A share that's browseable can be seen through a Microsoft Windows Network Neighborhood or My Network Places. Clients can't mount a share if **available = no**.

If you feel the need to limit access to specific users, the variables you see when you click the Advanced View button are interesting. They also support customizing file locks, so multiple users aren't allowed to write to the same file simultaneously; and file naming options, which help you navigate the difference between the case-sensitivity of Linux and the case-insensitivity of Microsoft Windows.

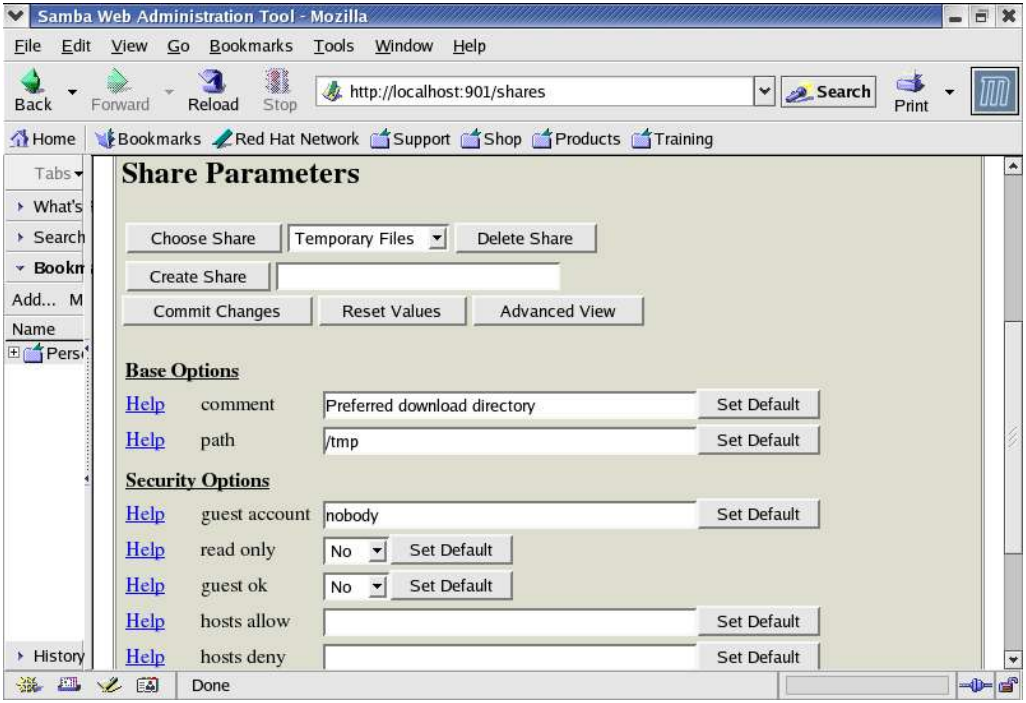


***Figure 14**. Configuring shares through the SWAT Shares menu.*

The kernel released with Red Hat Linux 9 does not fully support Access Control Lists (ACL). But when you examine the Advanced View variables, you can see that Samba is set with **nt acl support** by default. In other words, the permissions associated with a Linux file are automatically translated to a Windows NT-style ACL. I discuss Linux permissions and their relationship to ACLs in Chapter 4, "Setting Up Your File Server's Users."

SWAT Printers menu

Now that you've examined the SWAT Shares menu, don't forget the other part of a File and Print server. Click Printers. After you install a printer (see Chapter 7, "Configuring Printers"), you can use this menu to share that printer. Select it from the Choose Printer drop-down box shown in **Figure 15** and then click Choose Printer. You'll then be able to set parameters similar to those associated with sharing a directory.

The options in this section are fairly straightforward. Assuming you want users to see this printer in a Windows Network Neighborhood or My Network Places, you'll want to set the following variables:

```
printable=yes
available=yes
```



***Figure 15**. The SWAT Printers menu.*

While you can set up many different printing systems for Linux, the default is **cups**, which I describe in some detail in Chapter 7. If you're a Linux expert and are more comfortable with a traditional print system, the easiest-to-configure alternative is **lprng**, which is associated with the Line Print Daemon (which was the default through Red Hat Linux 7.3).

The SWAT configuration wizard

If you don't have any special directories or printers to configure, you can set up your Linux computer on a Microsoft Windows network with the SWAT configuration wizard. Click Wizard on the main SWAT menu. You'll see three essential items that you need to specify to configure your Linux computer on a Microsoft Windows network:

- Select a server type. A Stand Alone server is normally associated with a Workgroup; a Domain Member is a server that's not a controller on a Domain; and a Domain Controller includes the authentication database for the network.
- Choose a WINS configuration. If you use a WINS server on your network, you can set the local computer as a WINS server, or you can specify a remote WINS server for your network.
- You can enable users to have access to their home directories on the local computer.

For more information, click the Edit Parameter Values button. This opens a list of the key Samba variables as shown in **Figure 16**.

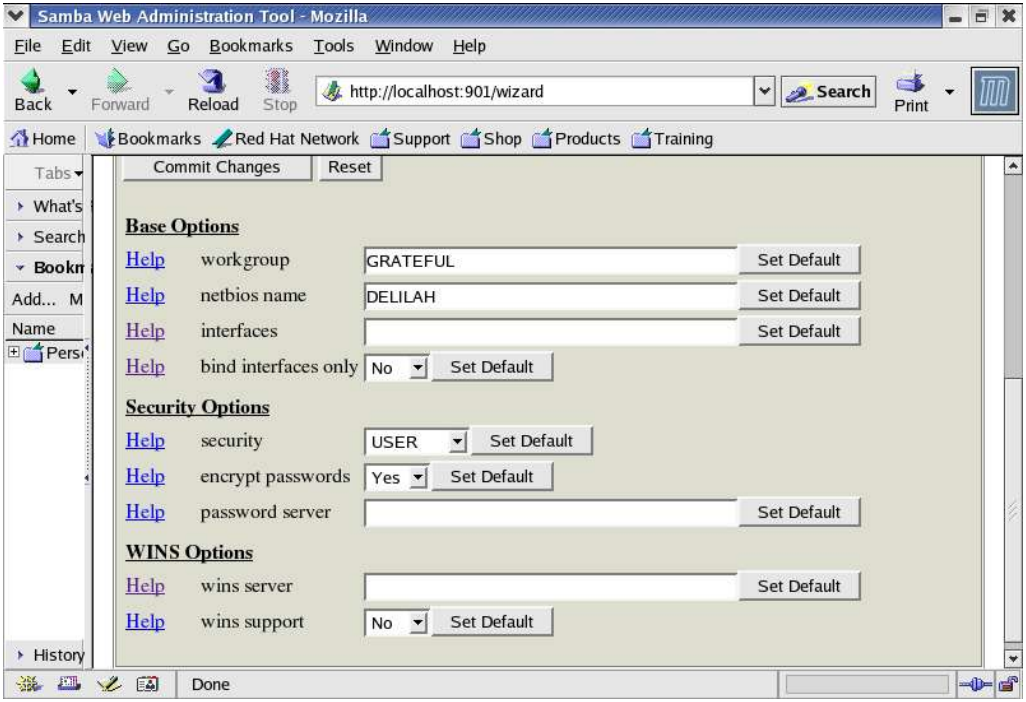


***Figure 16**. SWAT Wizard variables.*

While most of these variables seem familiar, it's worth a little trouble to summarize them in **Table 4**. Some of the variables in this table may seem repetitive, but it's useful to have this information in one place.

***Table 4**. Key Samba variables.*

Variable	Function
workgroup	The Microsoft Workgroup or Domain that you want this computer to join.
netbios name	Assigns a name to this computer for viewing in Microsoft Windows Network Neighborhood or My Network Places.
interfaces	Specifies the allowable network interface, such as eth0, or network, such as 192.168.20.; if you specify allowable networks, remember to add 127. to allow access from a local computer.
bind interfaces only	Set to yes if you want to limit access to certain interfaces.
security	Configures basic security options: **share**, **user**, **server**, **domain**.

Variable	Function
encrypt passwords	Enables encryption when passwords are sent over a network; associated with access from most Microsoft Windows computers (except Windows for Workgroups and early versions of Windows 95).
password server	If a remote computer (such as a PDC) is responsible for the user names and passwords on your network, specify it here.
wins server	If you have a WINS server for your network, specify its IP address or name.
wins support	If you want to set up the local computer as a WINS server, set this variable to yes. Don't do this if you already have another WINS server on your network.

SWAT Status menu

Once you've configured your server, you can maintain it with SWAT. Click Status to open the Status menu. In the menu shown in **Figure 17**, you can restart Samba to make it read any configuration changes that you've made. You can also monitor and cut off connections to your server.

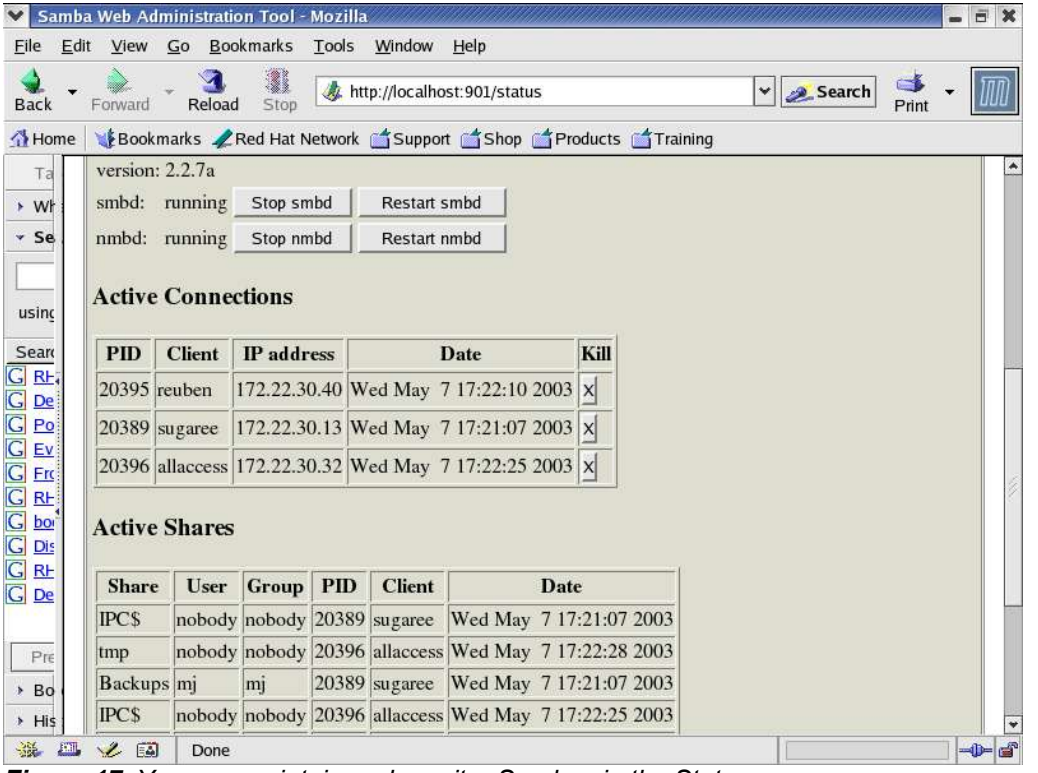


***Figure 17**. You can maintain and monitor Samba via the Status menu.*

Whenever you change the Samba configuration, the changes don't take effect until you restart the Samba server. On the Status menu, click the "Restart smbd" button. Anyone who's currently connected to your computer is cut off and will need to reconnect.

As you can see under Active Connections, three client computers are currently connected to my Samba server: **reuben**, **sugaree**, and **allaccess**. As an administrator, you can cut off access by clicking the X button in the Kill column. Under Active Shares, you can monitor the directories mounted from those computers, as well as their associated user names. Farther down the menu, you can check the files opened over the network.

SWAT View menu

Once you've changed the Samba configuration, you may want to review the Samba configuration file, smb.conf. You can do this from the command-line interface, or you can just click View atop the SWAT menu. This opens a view of smb.conf. Click the Full View button if you want to review all Samba variables in smb.conf.

SWAT Password menu

Before users can connect to a Samba server over a Microsoft Windows network, they need a user name on your computer. Click Password to open the Password menu. As you can see in **Figure 18**, this includes two submenus.



***Figure 18**. Using SWAT to manage passwords.*

In the Server Password Management submenu, you can manage the Samba users associated with the local computer. You can add, delete, disable, or enable a user; or you can change the password of an existing user.

In the Client/Server Password Management submenu, you can change the password on a remote server, even if it's a PDC. The entries shown in Figure 18 are self-explanatory; for example, Remote Machine corresponds to the name of the computer with the user name/password database that you're trying to change.

Now it's time to explain something a bit more difficult: the innards of the Samba configuration file, /etc/samba/smb.conf. Please read this file, especially if you're not comfortable editing text configuration files. It's important to understand the contents of this file even if you primarily use the Red Hat Samba Configurator or SWAT to configure your computer as a Samba server.

Configuring Samba in detail

The Red Hat Samba Server Configurator and SWAT are essentially just "front ends." In other words, your selections in these GUI tools essentially serve as intelligent editors for the smb.conf configuration file.



Appendix B includes sample smb.conf configuration files for a Domain member server and a PDC.

There are more than 300 different Samba variables; it's instructive to examine the smb.conf file that comes with the **samba-common** RPM package. It includes most of the variables that you might want to configure, and most of the variables include helpful comments —helpful, that is, to people more familiar with Linux.

The smb.conf file that I analyze in this section is based on the version of Samba that comes with Red Hat Linux 9. Different versions of Samba may include a default smb.conf file with different variables and comments.



*If you want to restore the original smb.conf and lmhosts files from the **samba-common** RPM, find the RPM. Just make sure it's the right RPM; otherwise you could overwrite a newer version of Samba. If you're using the original from Red Hat Linux 9, insert the appropriate Red Hat Linux 9 CD (#1). Once the CD is mounted, run the following commands:*

```
# cd /
# mount /mnt/cdrom
# rpm2cpio /mnt/cdrom/RedHat/RPMS/samba-common* | cpio -ivdu
```

Normally, the smb.conf configuration file is divided into two major sections: global parameters, where you configure the capabilities of your Linux computer as a server on a Microsoft Windows network; and shares, where you define how you're going to share the installed directories and printers on your network.

There are two basic ways to start a comment in a Linux configuration file: with a hash mark (#) and a semicolon (;). For example, the following two lines of code are both comments that are not processed by Samba:

```
#   password server = *
;   password server = <NT-Server-Name>
```



*Some Samba commands in various configuration files may look like they are misspelled. In fact, the Samba designers allow for different spellings. For example, **browsable** and **browseable**, and **writable** and **writeable**, work just as well.*

Opening smb.conf in a GUI

The most important files in Linux are the configuration files. Because they are all in text format, you should learn to edit them with a text editor. To open the GNOME text editor, gedit, click Main Menu | Accessories | Text Editor. Click the Open button, enter **/etc/samba/smb.conf** in the text box, and then click OK. You can now edit the Samba configuration file in the GUI as shown in **Figure 19**.

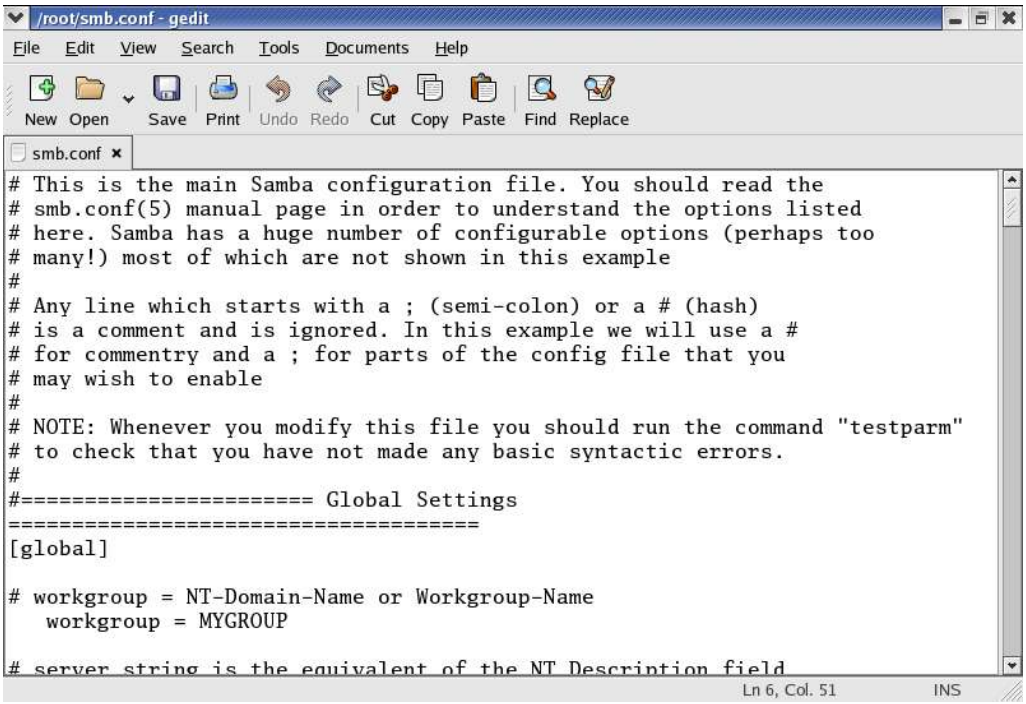


***Figure 19**. Editing the Samba configuration file.*

Global settings

In this section, I'll explain the global options available from the standard smb.conf configuration file. As you'll see, many of the options are commented, and can be activated by deleting the hash mark or semicolon at the beginning of the line. The following tag tells Samba to look for global settings:

```
[global]
```

Next, you'll want to identify the network that you want this computer to join. As noted in the comment, it doesn't matter whether you're joining a Windows Workgroup or Domain; just substitute the name of the Workgroup or Domain for MYGROUP.

```
# workgroup = NT-Domain-Name or Workgroup-Name
   workgroup = MYGROUP
```

If the **workgroup** variable is missing from the file, Samba assumes that you're joining the default Workgroup, which happens to be the default associated with Microsoft Windows peer-to-peer Workgroups: WORKGROUP. Next, you can customize the comment that other users see when they browse through Network Neighborhood or My Network Places.

```
# server string is the equivalent of the NT Description field
   server string = Samba Server
```

As described earlier, you can limit access to specific networks. The following commands, if active, limit access to computers with a network IP address of 192.168.1.0 and 192.168.2.0, as well as the loopback address (127.0.0.1).

```
;   hosts allow = 192.168.1. 192.168.2. 127.
```

The dot on the right side of the IP address acts as a wildcard. Remember, a command with a semicolon in front is a comment; if you want to use the command shown, open smb.conf in a text editor and delete the semicolon.

You could set up limits in the allowed IP addresses; for example, the following command allows access for all computers with a network address of 10.0.0.0, except those with a network address of 10.44.0.0:

```
hosts allow = 10. EXCEPT 10.44. 127.
```

Next you'll see how Samba finds and shares the list of installed printers on your system. From the commands shown, Samba gets its list from /etc/printcap, and automatically shares these printers on the Microsoft Windows network.

```
   printcap name = /etc/printcap
   load printers = yes
```

For Red Hat Linux, /etc/printcap includes your list of installed printers whether you're using the default CUPS or the former default LPRng print services. You'll see how this works in Chapter 7, "Configuring Printers."

```
printcap name = cups
```

As you can see from the comments, Samba can work with a number of different print systems. Unless you're configuring Samba for Unix or another alternative to Linux such as the Berkeley Standard Distribution (BSD), **cups** and **lprng** are the only systems that you need to know.

```
# It should not be necessary to spell out the print system type unless
# yours is non-standard. Currently supported print systems include:
# bsd, sysv, plp, lprng, aix, hpux, qnx, cups
   printing = cups
```

If you're setting up Samba on a pure peer-to-peer Workgroup, there is no central server for user names and passwords. Directories that you share are protected only by passwords. But on a Linux computer, you still need a user. By default, that user is **nobody**, which you can find in a Linux authentication list in the /etc/passwd file. Naturally, if you want to set up a separate account for a user named **pcguest**, you could activate this command:

```
;  guest account = pcguest
```

You'll also need to create the **pcguest** user with a command such as **useradd**, which I describe in Chapter 4, "Setting Up Your File Server's Users." One use for this type of guest account might be a home directory of public information.

When I have problems with Samba, the first place I look is in the log files. By default, the following command configures a separate log file for each client computer that connects to your Samba server:

```
   log file = /var/log/samba/%m.log
```

%m is a variable that is associated with the client "machine name." In Samba, this corresponds to the NetBIOS name of the client computer. For example, because one of my computers has a NetBIOS name of **StStephen**, Samba creates a connection log file named ststephen.log in the /var/log/samba directory.

If you have big log files, you may want to limit the amount of data in each file. While the default shown means that there is no limit, the default limit is 5000KB.

```
   max log size = 0
```



Log file limits generally are not needed in Red Hat Linux. By default, log files are rotated every week. The old log file is saved with a numeric extension such as .1, .2, .3, and .4, for up to four weeks of archived logs.

Earlier in this chapter, I covered the possible values for the **security** variable. Just to remind you, the options are **share**, **user**, **server**, and **domain**:

```
   security = user
```

If you've specified **security = server** or **domain**, you're passing along a user name and password request to a different server, presumably a PDC. You'll then want to name the PDC in some way by activating one of the following commands:

```
#   password server = My_PDC_Name [My_BDC_Name] [My_Next_BDC_Name]
#   password server = *
;   password server = <NT-Server-Name>
```

The first command can search through the named PDCs or even BDCs (Backup Domain Controllers). As of this writing, those would be Microsoft Windows server computers. If you want a backup to a Linux PDC, read the overview of backups in Chapter 9, "System Backup." If you're setting up a Samba Domain member server, you can use this command to look through the user name and password database on Microsoft PDCs and BDCs.



Once you've set up Samba as a PDC, backup controllers are rarely necessary. Linux is so much more reliable than Microsoft Windows that reboots are rarely required. If you absolutely have to have a BDC, read the unofficial Samba BDC HOWTO document at ***us3.samba.org/samba/ftp/docs/htmldocs/Samba-BDC-HOWTO.html****.*

The second command searches through the computers on the network for a Domain controller with the authentication database. The third option names a server, which does not have to be a Domain controller.

One of the problems with making Linux work in a Microsoft Windows network is the case-sensitivity of passwords and user names. Linux passwords and user names are case-sensitive. Microsoft Windows 9x/ME passwords and user names are not. The following commands, if active, check the uppercase and lowercase versions of passwords and user names of up to eight characters:

```
;  password level = 8
;  username level = 8
```

Activating these commands can affect the performance of your network. It means that uppercase and lowercase variations of each password and user name are checked each time someone logs in to your server.

As discussed earlier, passwords are encrypted by default. The user names and passwords that someone may use to log in to your Samba server (or PDC) are normally stored in a specific file, as documented by the following commands:

```
encrypt passwords = yes
smb passwd file = /etc/samba/smbpasswd
```



If you have to have computers running Windows 95 (before OSR2), Windows NT 4 Workstations (before Service Pack 3), or Windows for Workgroups (3.11) on your network, set ***encrypt passwords = no****. You'll also need to deactivate encryption on all other computers on your network. Alternatively, you can install Service Pack 2 (or later) to enable encryption on pre-OSR Windows 95 computers.*

Some Samba RPM packages (not including the version associated with Red Hat Linux 9) are installed with Secure Socket Layer (SSL) support. If you get SSL errors, activate the following command:

```
;   ssl CA certFile = /usr/share/ssl/certs/ca-bundle.crt
```

By definition, this file is a part of the **openssl** RPM package for secure communication on a network. The following commands allow you to set the parameters for changing Samba- and Linux-related passwords. The first command maintains the same password for logins to Samba shares and the Linux computer. This assumes that you have the same user name for both databases.

```
   unix password sync = yes
```

In Linux, **/usr/bin/passwd** ***username*** is the standard command for changing the password for a specific user. Samba substitutes the current user name for the **%u** variable.

```
   passwd program = /usr/bin/passwd %u
```

The next line works at the command-line interface; the **passwd chat** responses with the wildcards are part of what you would see at the command-line interface. To understand how this works, you really need to break this command into four components:

```
   passwd chat = *New*password* %n\n *Retype*new*password* %n\n \
*passwd:*all*authentication*tokens*updated*successfully*
```

- The ***New*password*** command is a prompt for a new password. The new password as typed is set as the value of **%n**.
- The ***Retype*new*password*** command prompts again for a new password; the entry is checked against the value of **%n**.
- The backslash (\) at the end of the first part of the line links the two code lines. In Linux, the backslash is said to "escape" the intent of the next character. In other words, when Samba reads this command, it ignores the Return character and adds the second line to the end of the first line.
- Finally, the command on the second line works as a response—assuming the passwords that you typed in matched. In Linux, the case of the password matters. In Samba, it may not. This is determined by three variables listed at the end of this section.

The wildcards support flexibility, and can be included in entries in the SWAT Password menu shown in Figure 18. As long as the **passwd** command works, you can delete the following command without any effect on your Samba server:

```
   pam password change = yes
```

You can set up two different databases of user names and passwords. In other words, you could use one user name to log in from a client on a Microsoft Windows network and a second user name to log in directly to a Linux computer. All you need to do is set up a database with the **smbadduser** command described in Chapter 4, "Setting Up Your File Server's Users." Then you can activate the following command:

```
;  username map = /etc/samba/smbusers
```

You can customize the Samba configuration file for different Linux and Unix computers on your network. The following command inserts the noted configuration file. Any commands from the inserted file supersede any active commands earlier in the file.

```
;   include = /etc/samba/smb.conf.%m
```

As described before, the **%m** variable corresponds to the NetBIOS name of the remote computer. For example, if you want a custom configuration file for a remote Linux computer named **nopaws**, this command would insert a file named /etc/samba/smb.conf.nopaws.

The following parameter in the default Samba configuration file should not matter. In other words, unless you have pre-OSR2 Windows 95 computers on your network, you're using encrypted passwords. In that case, there is no need for Pluggable Authentication Module (PAM) restrictions; thus, you can delete this command from your configuration file.

```
  obey pam restrictions = yes
```

You can regulate the speed of file transfer to and from your Samba server with the following command. The default is sufficient for most users; optimizing performance requires trial-and-error testing and is beyond the scope of this book.

```
   socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
```

Some Linux computers are connected to more than one network. If Samba is installed on such a computer, you can limit access to one or more network cards. If you activate this **interface** command, access is limited to the network cards with the given IP addresses. The **/24** is a subnet mask, in a notation known as Classless Inter Domain Routing (CIDR). The number corresponds to the number of bits; **/24** corresponds to a network mask of 255.255.255.0.

```
;   interfaces = 192.168.12.2/24 192.168.13.2/24
```

Alternatively, you can specify network interfaces by their device names; for example, the following command allows Samba communication only on the first Ethernet network card on the local computer. The second Ethernet card would be eth1, and so on.

```
interfaces = eth0
```

There may be different computers on various LANs that maintain browse lists for each network. If you want to synchronize your browse list with that of a remote computer, activate the following command, and replace it with the IP address of a remote browse master computer, or the broadcast address of a remote network.

```
;   remote browse sync = 192.168.3.25 192.168.5.255
```

If you activate this particular command, it synchronizes your browse list with that on the computer with an IP address of 192.168.3.25. It also sends a broadcast message to find and synchronize with the browse master on the network with an IP address of 192.168.5.0.

If users on other networks are having trouble finding your Samba computer, you can also tell others about your browse list. The following command, if you activate it, sends a broadcast message to a remote network with an IP address of 192.168.1.0, and a master browser with an address of 192.168.2.44.

```
;   remote announce = 192.168.1.255 192.168.2.44
```

By default, Samba servers participate in "elections" for browse master on a LAN. If you already have a preferred browse master such as a PDC, you should activate this command:

```
;   local master = no
```

Microsoft Windows LANs where the browse master is on a computer other than the PDC can have problems. If the PDC is on a remote LAN, this is not an issue.

If you want your computer to participate in browse master elections, you can give it a certain number of votes. The default **os level** is 20; if you've configured your Linux computer as a PDC, you should activate this command. The **os level** can be set as high as 255.

```
;   os level = 33
```

Browser elections proceed in two stages. The first stage depends on the operating system, as described in **Table 5**.

***Table 5**. Browser election values.*

Operating System	Value
Windows for Workgroups (3.11)	1
Windows 95/98/ME	1
Windows NT/2000/XP/2003	16
Windows NT/2000/2003 PDC	32

If there is a tie—for example, if there's more than one Samba server with an **os level** of 33, the election proceeds to a second stage. As shown in **Table 6**, the hierarchy of values for a browse election is highly dependent on its current role on the network.

***Table 6**. Additional browser values, in the case of a tie.*

Current Browser Role	Value
Current backup browser	1
Previous backup browser	2
Local master browser	4
Preferred master browser	8
WINS client	32
Domain master browser	128

While you can set an **os level** of up to 255, you should not do this unless it's on the computer that you're configuring as a PDC. And as described earlier, an **os level** of 32 is used by Microsoft servers to win a browser election as a PDC.

If the PDC and Domain master browser are on different computers, you'll observe errors such as browse lists not working on certain computers on the network. However, if you are configuring the local computer as the PDC and the Domain master browser, you should activate the following command:

```
;   domain master = yes
```

If you don't have a PDC on the LAN, you'll still need a master browser. In that case, you could set your computer as the preferred master browser by activating the following command:

```
;   preferred master = yes
```

If you are configuring the local server as a PDC and there are Windows 95/98/ME computers on the Domain, you'll want to activate this command. Remember, because **encrypt passwords = yes**, Windows 95 computers can't log in to this Domain unless the Microsoft Service Pack 2 (OSR2 or later) has been installed. And you need to set **security = user** (not **server** or **domain** as explained earlier in this chapter).

```
;   domain logons = yes
```

If you are configuring the local server as a PDC, you can set up logon scripts by NetBIOS name (**%m.bat**) or user name (**%U.bat**). Be sure to activate the appropriate one of the following two commands:

```
;   logon script = %m.bat
;   logon script = %U.bat
```

If you're converting from a Microsoft Windows PDC, you can use any scripts that you might have saved in directories such as the Windows NETLOGON share. If you want to configure a **logon script** variable by computer (**%m**), activate the first command, and set up scripts by NetBIOS name. If you want to configure a **logon script** variable by user (**%u**), activate the second command, and set up scripts by user name. You can define the directory for these scripts under the [netlogon] tag later in this file.

When you convert from a Microsoft Windows PDC on Windows NT 4, you should be able to copy the appropriate files directly from the Windows PDC NETLOGON share, which is commonly the C:\WINNT\system32\Repl\Import directory.

As most Microsoft Windows administrators already know, a roving profile provides a consistent look and feel for users who log in from different workstations on a network. You can store roving profiles on a Samba server. For example, if you activate the following command, logon profiles are stored as defined by the [profiles] tag later in the smb.conf file.

```
;   logon path = \\%L\Profiles\%U
```

In this case, the **%L** also defines the NetBIOS name of the computer, and the **%U**, once again, defines the user name.



To understand where profiles are stored, read the commentary associated with the [profiles] tag later in this section.

If you want to set up a WINS server on this computer, activate the following command. Just make sure that there is no more than one WINS server on the LAN. Naturally, you'll need to substitute the actual IP address of the server for w.x.y.z.

```
;   wins server = w.x.y.z
```

Modern Microsoft operating systems and Linux with an active Samba server can all be configured to access a WINS server. However, if you have a Linux client without the Samba server software, you'll need to activate the following command. Needless to say, this requires a configured WINS server on the network.

```
;   wins proxy = yes
```

The computers on a network may not always make it into a WINS server database. If you have a DNS server and want to use it as a backup, change the following default from no to yes.

```
   dns proxy = no
```

As described earlier, one of the major issues associated with Samba is the difference in case-sensitivity between Linux and Microsoft Windows. By default, the case of file names is preserved. If you don't like this default, activate the following command:

```
;  preserve case = no
```

A more specific issue is what happens to older DOS 8.3-style file names such as DOCUME~1.DOC. By default, case is preserved. If you want to set a different case, activate this command:

```
;  short preserve case = no
```

If you've activated one of the previous commands, you can set a **default case** variable for file names that are transferred. This command corresponds to the default; if you prefer, you can change "lower" to "upper":

```
;  default case = lower
```

Linux is normally case-sensitive. Samba is not, as defined by the following command:

```
;  case sensitive = no
```

If you set this variable to yes, file names such as Document.doc are considered different from document.doc.

As you can see, you can define a broad range of Samba parameters for the global system. The following sections illustrate how to share directories and printers on a Samba server.

Defining shared directories and printers

The shares defined in the default Samba configuration file address a wide variety of situations. The following sections analyze each of these share settings in detail. You can then activate and customize these settings to meet your needs. I define additional scenarios for shared directories in Chapter 4, "Setting Up Your File Server's Users," and for shared printers in Chapter 7, "Configuring Printers."

Samba includes a default share that allows users to access their home directories on your Linux computer. It allows you to configure shared directories for network logons and roving profiles. It also supports shared printers with varying permissions.

Shares in smb.conf are organized in *stanzas*. A stanza in a Linux configuration file is a group of commands organized together for one purpose. For example, the following stanza relates to the way that Samba allows Linux to share user home directories:

```
[homes]
   comment = Home Directories
   browseable = no
   writable = yes
   valid users = %S
   create mode = 0664
   directory mode = 0775
# If you want users samba doesn't recognize to be mapped to a guest user
; map to guest = bad user
```

Home directories

By default, every user who logs in to a Linux computer is taken to his or her home directory. Users who connect to a Samba server—and who have an account on that Linux computer—have default access to the same home directories. The settings are associated with the [homes] tag in the Samba configuration file.

This stanza first starts with a **comment** variable, viewable in Windows Network Neighborhood or My Network Places. Modify it as desired.

```
   comment = Home Directories
```

By default, you can tell Samba to keep home directories of individual users off the browse list by using the following command.

```
   browseable = no
```

The directory is still browseable if the user logs in with the right Linux user name and password. If you change this variable to **browseable = yes**, Samba adds the home directories of all users on your Linux computer to the browse list.

Generally, you'll want to allow users to have write access to their home directories. But there are exceptions; you can delete or disable user accounts as described in Chapter 4, "Setting Up Your File Server's Users."

```
   writable = yes
```

With home directories, you want to limit access to the owner. On a Linux computer, the name of a home directory such as /home/mj corresponds to the name of the user (**mj**). As the following command sets the **valid users** variable to the name of the share (**%S**), it's setting it to the name of the owner of the home directory.

```
valid users = %S
```

All Linux files have permissions associated with the file owner, group owner, and all others on the computer. The **create mode** command specifies the permissions for files that you copy, in this case, to the home directory on the Linux computer. I explain these permission numbers in detail in Chapter 4, "Setting Up Your File Server's Users."

```
create mode = 0664
```

As with files, all Linux directories have permissions. This command assigns permissions for all directories that you copy or create:

```
directory mode = 0775
```

Unless the computer is on a Domain, anyone without a valid account on the Linux computer won't be able to log in to a home directory. If you activate this command, users that aren't on your Samba system are automatically mapped to the **guest** user, as defined earlier in the Samba configuration file with the **guest account** variable described earlier in this section.

```
; map to guest = bad user
```

If you're configuring a Linux computer as a Domain member server, users are mapped to a home directory even if it doesn't exist. For example, assume Whil has an account on the PDC, but has no account or /home/whil home directory on the member server. The default share makes it look like Whil has a home directory on the member server.

Therefore, you may want to deactivate the Home Directories stanza for Linux Domain member servers.

Domain network logons

The next stanza, [netlogon], allows you to set up the Network Logon Service on a Samba server configured as a PDC. This assumes you've set the following commands earlier in this file:

```
security = user
domain master = yes
domain logons = yes
```

Then to set up the NETLOGON share for user logon scripts, you'll want to modify as needed, and then activate the following commands. Don't forget to activate the [netlogon] tag by removing the semicolon in front of the tag:

```
; [netlogon]
```

It's generally a good practice to add a comment to every network share; if the share is visible, the following comment identifies the share to your users.

```
;   comment = Network Logon Service
```

There is no required path for network logon scripts; feel free to change the **path** variable to the directory of your choice. Because a login to a Samba-based PDC directs a user to his or her home directory, many administrators replace this command with **path = /home/netlogon**.

```
;   path = /usr/local/samba/lib/netlogon
```

You may want to change this default to **guest ok = no** to prevent users without accounts from reading users' startup scripts.

```
;   guest ok = yes
```

If you don't want individual users to rewrite their logon scripts, activate the following command:

```
;   writable = no
```

Even though the following command is in the default Samba configuration file, there is no reason to activate it. If you do, it would disable file locks, which would then allow multiple users on different computers to read the same file simultaneously.

```
;   share modes = no
```

There's no need for users to browse through the [netlogon] share. Crackers could use it to find a list of users on your Domain. If crackers could access the scripts in this directory, they could create all sorts of trouble for your system. Thus, I recommend that you add one more command to prevent browsing:

```
browseable = no
```

Roaming profiles

Users who log in from different workstations on a Domain often prefer the same look and feel on different computers. This is where roaming profiles are useful.

In Microsoft Windows, the user part of a registry can be stored in a file on a remote server. As Windows administrators know, the file is normally USER.DAT for Windows 9x/ME computers and NTUSER.DAT for Windows NT/2000/XP computers. When that user logs in to the Domain, the appropriate file is downloaded to the current workstation.

We won't use much of the standard [profiles] stanza from the default smb.conf configuration file, because the following commands, if active, would leave profiles vulnerable to guests and other users on your Domain:

```
;    path = /usr/local/samba/profiles
;    browseable = no
;    guest ok = yes
```

Instead, we'll activate the following commands to configure a shared profiles directory. First, as with other shares, you need to list the stanza in the configuration file:

```
[Profiles]
```

You can add a comment to further identify the share to administrators, such as:

```
   comment = Roving Profiles
```

Naturally, you'll want to define the directory **path** variable for the profiles, which works with the **logon path** command that you set earlier. For example, with a **logon path = \\%L\Profiles\%U** and the following command, you can set the **path** variable for your profiles:

```
path = /home/profiles
```

Don't forget to create the directory as defined by the **path** variable (/home/profiles); you'll want to give all users full permissions to that directory with the following commands. (Don't include these commands in the Samba configuration file; run them from the command-line interface.)

```
# mkdir /home/profiles
# chmod 777 /home/profiles
```

Windows NT/2000/XP NTUSER.DAT files are stored in the /home/profiles/%U directory, where **%U** corresponds to the user name. For example, with the commands shown in this section, the profiles for user **mj** are stored in the /home/profiles/mj directory. Unless the profile is mandatory, you need this to be a writable directory for the subject user, which you can set with the following command:

```
read only = no
```

Microsoft Windows users often do change their profiles. To allow the owner of the profile file to write his or her profile directory, you'll need to set the following permissions for files and the directory:

```
create mode = 0600
directory mode = 0700
```

As with the [netlogon] share, it's best to keep [profiles] off the browse list, and prevent access to guest users:

```
browseable = no
guest ok = no
```

A common directory

You may want a common directory for all of the users on your network. It's a common practice to use the /tmp directory to share files and store downloads. If you intend to use /tmp for either of these purposes, you may want to mount it on a separate partition when you install Linux.

The default Samba configuration file provides standard commands that you can use to share the /tmp directory with all users. Remember, to activate these commands, delete the semicolon at the start of the line. Naturally, the stanza starts with a tag name:

```
;[tmp]
```

As with other shared directories, the comment you enter is shown in Windows Network Neighborhood or My Network Places and can help your users understand the purpose of the share. Change the value of the **comment** variable as desired.

```
;   comment = Temporary file space
```

Naturally, if you want all users to be able to share files in this directory, you need to make it writable. The **writable = yes** command is synonymous to the following:

```
;   read only = no
```

And if you have guest users, you can make the directory accessible to them with the following command:

```
;   public = yes
```

A group directory

You can set up directories for individual groups of users. The following command stanza, if activated, creates a directory that's writable for a specific group of users, and read-only for everyone else who connects to your Samba server. As always, the stanza starts with a specific tag:

```
;[public]
```

Once again, the stanza follows up with a **comment** command to help explain the purpose of the share:

```
;   comment = Public Stuff
```

The path defines the shared directory. When you create this directory, access depends on Linux ownership of this directory. For example, if users **mj** and **ez** are part of the **staff** group, you need to make sure that **staff** is the group owner of the /home/samba directory—and that members of the **staff** group have permission to write to /home/samba. For more information on this process, see Chapter 4, "Setting Up Your File Server's Users."

```
;   path = /home/samba
```

You can set the directory and contents to be visible to all users who connect to your Samba server with the following command:

```
;   public = yes
```

Before any user can write to /home/samba, you need to include the following command:

```
;   writable = yes
```

The following command is probably not critical, unless you have a printer that might use this directory for print spool files:

```
;   printable = no
```

You can specify the users and groups who are allowed to write to this directory with the **write list** command. If you activate the following command, users in the **staff** group are allowed to write files to the /home/samba directory.

```
;   write list = @staff
```

Alternatively, you can specify a list of users in the write list, such as:

```
;   write list = mj ez
```

When a user writes a file to the /home/samba directory, Linux assigns file ownership to that user. For example, if user **mj** copies the Chapter3.doc file to /home/samba, only user **mj** gets to overwrite that file.

Sharing printers

You can't use Samba to configure a printer; you'll learn about this process in Chapter 7. However, you can use Samba to share installed printers, based on the list collected through the /etc/printcap file. Naturally, this starts with the name of the Printers stanza:

```
[printers]
```

Based on what you've seen so far, two of the commands from the default smb.conf configuration file used to matter but no longer make any difference. The **comment** command no longer drives the comment associated with shared printers in a Windows Network Neighborhood or My Network Places. In other words, you can delete the following command, and it'll make no difference:

```
   comment = All Printers
```

The standard Microsoft Windows and Linux print systems both work on spool files. In other words, they process a print job through a filter; the print job is processed into a file in the spool directory. In Samba, the spool directory is defined by the path variable, as shown:

```
   path = /var/spool/samba
```

While printers are not **browseable** to guest users, they are visible to authorized users in a Windows Network Neighborhood or My Network Places. Therefore, the following command does not matter unless you have guest users:

```
browseable = no
```

If you actually want to authorize guest users on your printers, change the following line to **guest ok = yes**.

```
guest ok = no
```

The print spool directory can be read-only. Despite the following line, your printer can still send jobs to the /var/spool/samba directory. In Samba, **writable = no** works exactly like **read only = yes**.

```
writable = no
```

The following command allows your printer to write jobs to the print spool directory, superseding the **writable = no** command.

```
printable = yes
```

A dedicated printer

It's possible to limit the use of a printer to one or more users. For example, if you activate the following commands, you can set up a printer for the sole use of the user named Fred. The first couple of commands should be familiar now; they include a tag for the stanza and a comment for the printer:

```
;[fredsprn]
;   comment = Fred's Printer
```

The **valid users** command specifies a list of users allowed to use the specified share—in this case, the printer. As with the **write list** command described in the earlier section "A group directory," you can create a list of users and/or groups. The following command limits use of this printer to **fred**; the print spool is also set to use Fred's home directory:

```
;   valid users = fred
;   path = /home/fred
```

Fred's home directory is a good choice, because by definition, Fred has write access to this directory. If you use a different directory, Fred needs write access to that directory as well.

Assuming you have more than one printer, you'll want to specify the printer assigned to Fred. Based on previous commands, the following command assumes that **freds_printer** is the name of a printer in the /etc/printcap file. The following command keeps others from seeing **freds_printer** in the browse list.

```
;   printer = freds_printer
;   public = no
```

The final two commands in this stanza are identical to those in the [printers] stanza, which allows the **valid user** to use this printer:

```
;   writable = no
;   printable = yes
```

A private directory

Users on your computer have their own private directories—their home directory. Many distributions other than Red Hat Linux organize users as part of the **users** group. In that case, other users can see and read the files in all /home directories.

In that case, special commands are required to give user **fred** a private directory. For example, you can activate the following commands to set up a private directory that is visible and accessible only to **fred**:

```
;[fredsdir]
;   comment = Fred's Service
;   path = /usr/somewhere/private
;   valid users = fred
;   public = no
;   writable = yes
;   printable = no
```

For these commands to work, you need to create the directory specified by the **path** variable. You need to set **fred** as the user and group owner of the /usr/somewhere/private directory.

Directories by client

Just as the [homes] share configures directories by user, you can set up special directories based on the NetBIOS name of the client computers on your network. For example, one of my computers has a NetBIOS name of **StStephen**, so I've set up a home directory for that computer of /home/ststephen. I've modified the **path** command from the default Samba configuration file, and configured NetBIOS client directories in the following stanza:

```
[pchome]
  comment = PC Directories
  path = /home/%m
  public = no
  writable = yes
```

Remember, the **%m** variable corresponds to the NetBIOS name of a computer. Thus, the /home/ststephen directory is visible only to users who log in to your Samba server from the **StStephen** computer.

When you create the specific directory, you'll need to make it accessible to all users. Otherwise, some users may not be able to write to that particular NetBIOS name-specific directory. As described in Chapter 4, "Setting Up Your File Server's Users," you can do that with the following command from the command-line interface:

```
# chmod 777 /home/ststephen
```

All users as guest

You can set up a public directory for all users. The following stanza sets up a guest directory where everyone can overwrite everyone else's files. I've changed the **path** variable to be a part of the /home directory.

```
[public]
   path = /home/public
   public = yes
   only guest = yes
   writable = yes
   printable = no
```

The key command is **only guest = yes**, which directs all logins to the guest account. No matter how you log in to this shared directory on the Samba server, any files you create are owned by the guest account. Anyone else who logs in gets that same guest account. Therefore, anyone can delete files that others create or copy into this directory.

A small group

The final stanza allows you to set up a private directory for a group of two users. It's similar to the directory described in the earlier section "A group directory," except this directory is private. You still need to set up a separate directory owned by a group that includes the desired users.

You then need to set the shared directory with complete group permissions, which in this case permits access only to **mary** and **fred**. Yet each user retains individual ownership of any files he or she places in the shared directory. For example, if Fred creates the chapter4.doc file, Mary can't delete that file.

```
[myshare]
   comment = Mary's and Fred's stuff
   path = /home/maryfred
   valid users = mary fred
   public = no
   writable = yes
   printable = no
   create mask = 0765
```

Conclusion

You can configure Samba to set up your Linux computer as a client, a server, or a Domain controller on a Microsoft Windows network. But before doing so, you need to understand the background behind the Microsoft CIFS. In this system, you can set up client and server computers in Microsoft networks, organized in Workgroups and Domains.

There is an alternative; you can set up Microsoft Services for Unix on Microsoft Windows computers. Unfortunately, this is costly and it does not support Microsoft Windows 9x/ME computers on your LAN. In contrast, Samba is free and reliable and is therefore the main focus of this book.

Before you can configure Samba, you need to understand how to start and manage the Samba process. Samba users depend on the basic organization of users on your Linux

computer. In Red Hat Linux, there are several basic Samba configuration files in the /etc/samba directory.

There are two GUI tools that you can use to configure Samba. Red Hat has created a Samba Server Configurator, which can help you set up a Samba server with basic settings. There's also a more complete utility known as the Samba Web Administration Tool (SWAT).

But to really understand how Samba works, you need to get into the nitty-gritty of the main Samba configuration file, smb.conf, which includes global and share parameters. I've analyzed the default version of this file in detail, so you can more intelligently decide how to configure Samba on your computer and network.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 4
Setting Up Your File Server's Users

Computer networks revolve around users and groups. Users can log in to computers and networks, and they can access the files and directories that they own. In a Microsoft-based network, administrators usually share additional files and directories with groups. While the Microsoft and Linux user and group databases are different, this chapter shows how to make them work together. When you're finished with this chapter, you'll be able to set up users and groups on a Microsoft or a Linux-based PDC.

If you're going to set up a Linux PDC, you need to know how to set up Linux users and groups in some detail. As described in Chapter 3, you also need a corresponding Samba user database on that Linux PDC to allow access from Microsoft Windows computers.

When you set up a Domain, you're configuring a single database of user names and passwords. You can configure the **winbind** daemon to allow Linux computers on a network to work with a Microsoft Windows PDC. Alternatively, you can configure a Linux computer with Samba as a PDC with logon scripts, profiles, and client computer accounts customized for Microsoft Windows workstations.

Users who log in to a Linux PDC normally get access at least to their home directories. Additional shares are defined in the Linux PDC's smb.conf file and are limited by the Linux model for file and directory ownership and permissions. Users who connect to a Linux member server also can connect to shares as defined by the smb.conf file associated with each member server.



You can find sample PDC and member server smb.conf configuration files in Appendix B.

Although Samba does not fully support Microsoft's access control limits, you can still configure access controls for custom groups on the directories of your choice. This chapter builds on the smb.conf settings described in Chapter 3, "Setting Up Your Server File System." On larger networks, it's important to regulate the resources used by each user and group; I'll show you how to make this happen with the Linux quota system.

Because this is a chapter for administrators, almost all of the following commands and utilities require access to the administrative account on the Linux computer that you're setting up as a PDC or a member server. In other words, after logging in as a regular user, run the following command to start the Linux administrative account; enter the root user's password when prompted.

```
# su
password:
```



Linux does not provide as fine-grained default administrative control as the Microsoft Windows server operating systems. There is only one default Linux administrative user: root.

Users and accounts

In this section, I'll show you how to set up and configure Linux users and groups in detail. You can use the Red Hat User Manager to create and manage users and groups. More experienced Linux users may prefer to use command-line utilities. Whatever method you choose, Red Hat Linux users and groups get encrypted passwords. You can regulate the use and life of each password. If necessary, you can disable specific users.

Therefore, before you can set up Linux with Samba as a server on a Microsoft Windows network, you need to set up a simulated Microsoft Windows user and password database. This process was described briefly in Chapter 3. Finally, if you're setting up a Linux PDC, you'll also need to set up computer accounts in the simulated Microsoft Windows database.

Linux user and group accounts

In Chapter 3, I described the three basic types of group accounts: administrative, accounts associated with services, and accounts for individual users. You can organize users into groups of your choice. You may have already created one regular user with the First Boot utility described in Chapter 2, "Installing Linux as a File Server."

You can create and manage users and groups with various Linux utilities at the command-line interface, or you can use the Red Hat User Manager. You'll learn how to create users and groups with different IDs. I'll also show you how to manage the passwords and the allowed activity associated with each account.

The Red Hat User Manager

Red Hat has recently created a GUI tool for managing users and groups, known as the Red Hat User Manager. To start it from the GUI, click Main Menu | System Settings | Users and Groups. See **Figure 1**.

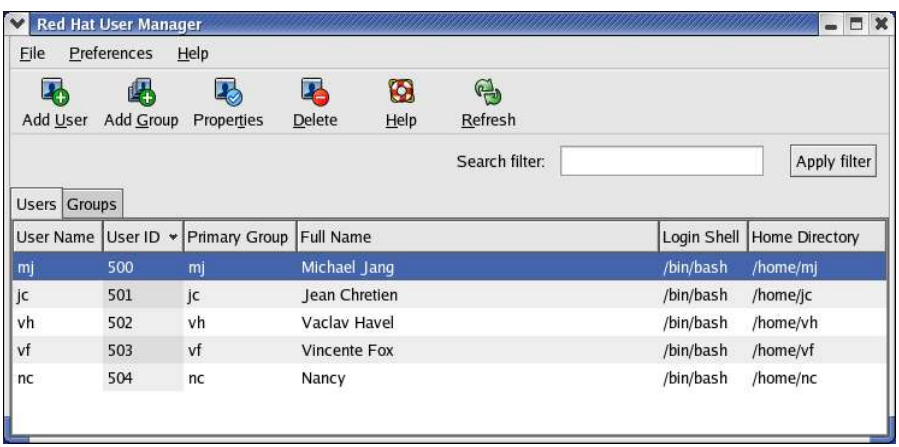


***Figure 1**. The Red Hat User Manager helps you to manage users and groups.*

If you're going to create a user name and password authentication database on a Linux computer, you need to know how to create and regulate users and groups. I'll start by showing you how to create an account for a user named Microsoft User.

1. Click Add User. This opens the Create New User window shown in **Figure 2**.



***Figure 2**. Creating a new user account.*

2. Configure the new user as prompted. Refer to **Table 1** for guidance on each entry.

***Table 1**. Creating a new user.*

Entry	Description
User Name	The login name; Linux supports user names up to 32 characters. Blanks and punctuation marks are not allowed.
Full Name	Enter the information of your choice in this text box; it functions as a comment for this user.
Password	Enter your intended password for the user.
Confirm Password	Repeat the intended password; a user can change his or her password with the **passwd** command.
Login Shell	The command interpreter at the text prompt; the default shell, /bin/bash, is most common. Don't change unless this user desires one of the other command interpreters available for Linux. Alternatively, if you're configuring an account on a Linux server for a user who will never log in directly into a Linux workstation (that is, Windows-only users), you can set this to /sbin/nologin.
Create home directory	If you're creating a regular Linux account, you'll want a home directory for that user.
Home Directory	By default, this is /home/%U, where **%U** corresponds to the user name. You can set a different home directory if desired.

Entry	Description
Create a private group for the user	By default, Red Hat Linux sets up private groups for all regular Linux users. Unless you have a good reason, keep this default; it helps secure the files on individual home directories.
Specify user ID manually	By default, Red Hat Linux assigns user IDs sequentially from 500. If it's a special type of account, such as for a client computer, you may want to set the number manually to a number up to 65,535.
UID	User ID; if the UID box is not grayed out, you can assign the UID number of your choice. Be careful; don't enter a UID already assigned to someone else. See the following note for more information on User ID numbers.



In Linux, every user gets an ID number. The administrative user, root, gets a User ID (UID) of 0. Other service users get UIDs between 1 and 99. In Red Hat Linux, users get ID numbers of 500 and above, numbered consecutively.

3. Click OK to complete the process. You should see the new user in the main list. If you've accepted the default to create a private group for the user, you'll also see a new entry on the Groups tab.

You can do more to configure a user account. Highlight the user that you just created, and then click Properties. This opens the User Properties window shown in **Figure 3**. It includes four tabs of settings for an account described in **Table 2**.



***Figure 3**. Configuring a user account.*

***Table 2**. Configuring a user account.*

Tab	Settings
User Data	Includes the basic information that you already configured for this account: user name, password, home directory, and login shell.
Account Info	Allows you to disable the account, or let it expire at a certain date.
Password Info	Supports password aging. As shown in Figure 3, you can make this user change the password after a fixed number of days.
Groups	Permits you to assign this particular user to be a member of the groups of your choice.

Assuming you created the users sequentially, they should be members of their own group. For example, the user **mj** is by default the only member of the group named **mj**. The User ID and Group ID should be the same, as shown in **Figure 4**.

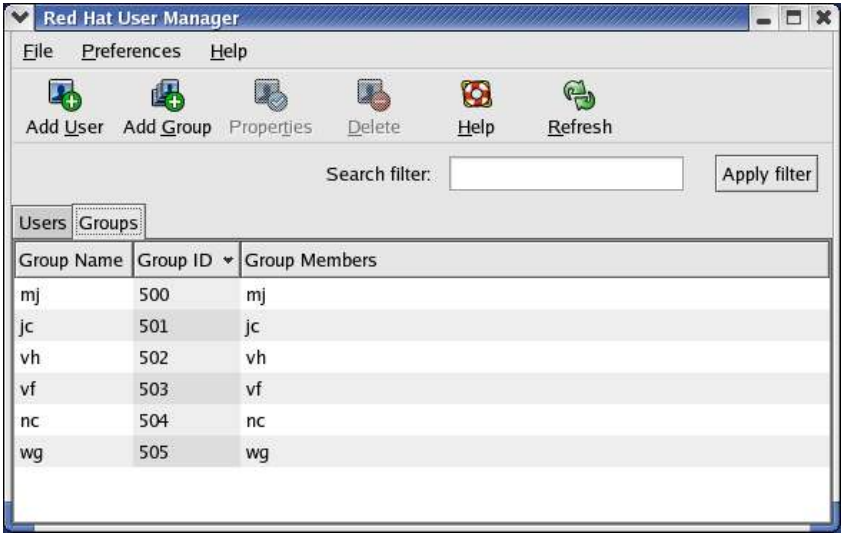


***Figure 4**. Standard Red Hat Linux private groups.*

Now you can create any special groups that you need in your organization. For example, assume users **jc** and **vf** want to set up a project to promote trade between Canada and Mexico, and are exchanging files on your system. You could set up a group named **canmex**, and make these users members of this group. One way to do so is with the following procedure:

1. In the Red Hat User Manager, click Add Group.
2. In the Group Name text box, enter the desired name for your new group, in this case, **canmex**.
3. Select the "Specify group ID manually" option. Set a Group ID (GID) outside the range of normal user IDs. For example, if you have 100 users on your system, your UID numbers won't exceed 600, and a GID of 1000 would work. Click OK to continue.

4. On the Group tab, highlight the **canmex** group and click Properties.
5. On the Group Users tab, you'll see a list of available users from your /etc/passwd file. Select the users of your choice; in this case, you'd select users **jc** and **vf**.

This private group is now ready for a shared directory, similar to what I described for users **mary** and **fred** in Chapter 3, "Setting Up Your Server File System."

Creating users and groups at the command line

As with other Linux GUI tools, the Red Hat User Manager is simply a front end to utilities that you can run from the command-line interface. In most cases, the command-line utilities are straightforward and faster than the GUI utilities. For example, if you wanted to add user **jc** to your system, you'd run the following commands as the root user:

```
# /usr/sbin/useradd jc
# passwd jc
Changing password for user jc
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
#
```

After you create a new user, use a text editor to open the files listed in **Table 3**. These files contain users, groups, and associated settings.

***Table 3**. User and group configuration files.*

File	Function
/etc/passwd	Main user configuration file; includes user name, user ID, group ID, home directory, and a login shell. In Red Hat Linux, this also includes a link to /etc/shadow for an encrypted password and aging information.
/etc/shadow	Secure user configuration file; includes user name, encrypted password, and password and account aging information.
/etc/group	Main group configuration file; includes group name, group ID, link to /etc/gshadow for the group password, and list of group members.
/etc/gshadow	Secure group configuration file; includes group name, encrypted password, group administrator, and group members.

Sometimes you might want to deactivate a user. Someone may go on leave, be suspended, or forget to pay their bill. You could disable users through the Red Hat User Manager, or you could simply replace the **x** in /etc/passwd. For example, the normal entry for user **mj** in /etc/passwd is:

```
mj:x:500:500:Michael Jang:/home/mj:/bin/bash
```

If you're having a problem with that user, open /etc/passwd in a text editor and replace the **x** with an asterisk (*****). If user **mj** tries to log in, even the correct password is not accepted.



Always save Linux configuration files in text mode. Many word processors normally save files in binary format (such as a DOC file), which Linux can't read. The next time Linux tries to read a binary file, the related process will fail. For example, if you saved /etc/passwd in binary format, you would not be able to log in to Linux.

Deleting a user is almost as easy as adding one. If you're fed up with user **mj**, the following command deletes the user and the /home/mj directory, including all files in that directory.

```
# /usr/sbin/userdel -r mj
```

If you forget the **-r** switch, the **userdel** command does not delete the /home/mj directory. If you want to add a special group, it is fairly easy with the **groupadd** command.

However, you would also need to edit the /etc/shadow and /etc/gshadow configuration files directly to add members to the specified group. For example, to set up the **canmex** group described earlier, take the following steps:

1. Run the following command to create the **canmex** group with an ID of 1000:

   ```
   /usr/sbin/groupadd -g 1000 canmex
   ```

2. Open the /etc/group file in a text editor. Add the user names of your choice to the end of the line that starts with the name of the group. Save the file and exit from the text editor. Based on the previous example, the line will probably read as follows:

   ```
   canmex:x:1000:jc,vf
   ```

3. Repeat the process with the /etc/gshadow file. Based on the previous example, the applicable line will probably read as follows:

   ```
   canmex:!!::jc,vf
   ```

Configuring Samba users

In Chapter 3, you learned how to set up Samba users with the Red Hat Samba Configurator. To summarize, Windows users with an account on a Linux computer can sign in to shared directories on that Linux computer with their Linux account. If you've enabled the **username map** variable in smb.conf, you can configure a database of Windows and Linux users in /etc/samba/smbusers, and their corresponding passwords in /etc/samba/smbpasswd.



The Windows and Linux computers know about each other with the help of the WINS server that you set up on the PDC. Just as you can set up a WINS server on a Microsoft PDC, you can also set up a WINS server on a Linux computer with Samba through the smb.conf file as defined in Chapter 3, "Setting Up Your Server File System."

I'll now analyze these files in more detail, with the help of the **smbadduser** and **smbpasswd** commands.

Configuring Samba users from the command line

You learned to set up Samba users with the Red Hat Samba Configurator in Chapter 3. Setting up Samba users is actually easier to do from the command-line interface. Once you configure Samba users and activate the **username map** variable in smb.conf, Microsoft Windows users will be able to log in to a Samba server configured as a PDC, or a Samba member server on a peer-to-peer network.

When you set up Windows users on a Samba server, you can set up two different passwords: one for users who log in directly to a Linux computer, and another for users who log in to a Microsoft Windows Domain.

If you want to let a user log in to your Linux computer, that user needs a Linux password. You can set it up through the Red Hat User Manager. Or for a new user named **egon**, you could run the following command, entering the desired password when prompted:

```
# passwd egon
Changing password for user egon
New password:
Retype new password:
passwd: all authentication tokens updated succesfully.
#
```

If you're just creating an account for logins from a workstation on a Microsoft network, all you need to do is create a corresponding Samba account by using the following command:

```
smbadduser Linuxuser:Windowsuser
```

You're prompted to enter the password that your users will enter when connecting through a Microsoft Windows network. For example, if you want to create a Samba account for the Linux user **egon**, run the following command and enter the desired Microsoft Windows password when prompted:

```
# smbadduser egon
Adding: egon to /etc/samba/smbpasswd
Added user egon
Adding: {egon = egon} to /etc/samba/smbusers
------------------------------------------------------------------------
Enter password for egon
New SMB password:
Retype new SMB password:
Password changed for user egon.
```

To see how this works, it's instructive to take a look at the smbusers and smbpasswd files from the /etc/samba directory. First, the smbpasswd file lists the Linux user names and their corresponding Samba passwords, in encrypted format:

```
mj:500:FE42CF3006363168930:9A9654CB7ABCFC5F32DD55:[UX          ]:LCT-3EBBC646:
vh:501:FE42CD7E6F503168930:9A96FD5F32DD389DDF2E55:[UX          ]:LCT-3EBBD348:
jc:502:AADEEAAD3B435B51404EE:31D6CF3C59D7E0C089C0:[UX          ]:LCT-3EC13DE7:
mary:503:FE42CF300636D7E63168930:9A96D54CB79DDF2E55:[UX          ]:LCT-3EBC5AF1:
egon:504:FE42CF300636D7E6930:9ABCFC5F32DD389DDF2E55:[UX          ]:LCT-3EBC5B02:
nc:505:FE42CF30067E4B53168930:9A96FBCFC5F32DD3855:[UX          ]:LCT-3EC12250:
```

The names in the first column are the Linux user names. The second column lists the corresponding User IDs. The long list of letters and numbers in the third column is the Samba password, in a format usable for logins from Windows 9x/ME computers. The fourth column (before the "[UX") is the Samba password, in a format usable for logins from Windows NT/2000/XP computers. But note that all of the users listed in the smbpasswd file aren't listed in the smbusers file:

```
# Unix_name = SMB_name1 SMB_name2 ...
root = administrator admin
nobody = guest pcguest smbguest
jc = jean
vh = vaclav
nc = nancy
egon = egon
```

The only users listed in the **smbusers** file are those with different user names on a Microsoft Windows computer. For example, **administrator** is the standard Microsoft administrative user name. I created a Samba password for user **mj** with the **smbpasswd** command, so Samba doesn't make it a part of the **smbusers** file.

In Chapters 5 and 6, you'll use these passwords to log in to a Microsoft Windows-based network from Linux and Microsoft Windows workstations.

Configuring computer accounts

Before you can connect to a Domain from a Windows NT/2000/XP computer, you need to set up a computer account. If the Domain is governed by a Linux PDC, you should add one more [global] line to the smb.conf configuration file:

```
add user script = /usr/sbin/useradd -d /dev/null -g 100 -s /bin/false -M %u
```

When you set your Microsoft Windows workstation to connect to a Domain, you'll get a chance to create a computer account, as described in Chapter 6, "Connecting Windows Workstations." This particular **add user script** command does the following:

- The **/usr/sbin/useradd** command adds a user with the noted switches.
- The **-d /dev/null** option sets a null home directory.
- The **-g 100** option assigns computer accounts to a group ID of 100.
- The **-s /bin/false** option sets up a false default shell, so anyone who breaks into this account is unable to run any Linux commands.
- The **-M** switch keeps the **/usr/sbin/useradd** command from creating a home directory.

- Finally, the **%u** option takes the NetBIOS name of the computer, adds a dollar sign ($), and then sets it up as the user name in the /etc/passwd file. The following is an example of a computer account line in the /etc/passwd file on the Linux computer that I've configured as a PDC:

```
daisy$:x:506:100::/dev/null:/bin/false
```

Because this is Samba, you'll also see a corresponding line in the /etc/samba/smbpasswd file, with the same user name and UID. If you're going to create additional accounts for regular users in the future, you may want to change the computer account UIDs to a different number range; for example, you could change the UID of 506 to 1506. If you do so, just remember to change the UIDs in both key files: /etc/passwd and /etc/samba/smbpasswd.

Naturally, you can also create computer accounts with the commands that I've shown you so far in this chapter, **/usr/sbin/useradd** and **smbpasswd**.

If your workstations are already connected to a Windows PDC and you just want to connect to a Linux-based PDC, there is an alternative. You can create individual computer accounts from the command-line interface. For example, you could create the aforementioned account for the computer with a NetBIOS name of **daisy** with the following commands:

```
# /usr/sbin/useradd -d /dev/null -g 100 -s /bin/false -M daisy$
# smbpasswd -am daisy
```



If you have a large number of computers on your network, it may be risky to add computer accounts directly from the command-line interface. Crackers can break into your network by logging in using a new computer account before you even connect your computers to the network.

Users on a Domain

In a network of Microsoft Windows and Linux computers connected by Samba, there are two possibilities for PDCs. You can have a PDC on a Microsoft Windows operating system, or a PDC on a Samba-enabled Linux computer. In either case, you can also configure additional Samba-enabled Linux computers as member servers. The way you set up Domain users differs depending on the configuration you select.

Using the Microsoft user database

If you've invested a lot of time in your Microsoft user database, you may not want to convert it to Samba, at least not right away. If you're implementing Linux with Samba on your networks for the first time, one approach is to set up a Samba-enabled Linux member server on a Domain with a PDC on a Microsoft computer.

As a member server, you want the Linux computer to use the Microsoft Domain-authentication database of user names and passwords. This is possible with the **winbind** service, which is installed with Samba if you're using Red Hat Linux. Setting up **winbind** on a Samba member server is a four-step process:

1. Edit the name service switch file, /etc/nsswitch.conf, so Samba knows to use **winbind** to read the authentication database from a PDC on a Microsoft computer.
2. Edit the main Samba configuration file, /etc/samba/smb.conf, to point to the PDC computer and assign appropriate user and group ID numbers to users from the PDC's authentication database.
3. Add the Windows administrator account to your Samba password database, /etc/samba/smbpasswd.
4. Start the **winbindd** daemon.

Afterwards, you can inspect the result in your /etc/passwd and /etc/group files with a couple of different commands. The following sections examine this process in detail.

Editing nsswitch.conf

Open the /etc/nsswitch.conf file in a text editor. Take a look at the following two lines:

```
passwd:     files
group:      files
```

These lines tell Linux to look to the standard files for the authentication database: /etc/passwd for users, and /etc/group for groups. It's simple to add **winbind** to this list; just change these lines to:

```
passwd:     files winbind
group:      files winbind
```

Close and save your changes to /etc/nsswitch.conf. Run the following command to activate the appropriate **winbind** program library:

```
# /sbin/ldconfig
```

Now the next time your Samba computer looks for a Windows Domain user name, it looks through your /etc/passwd file. If the user name isn't there, it next uses the **winbind** daemon to search through the authentication database on the Windows-based PDC.

Editing the main Samba configuration file

However, at this point, **winbind** doesn't know where to find the Windows-based PDC. Thus, the next step is to edit the main Samba configuration file. Open /etc/samba/smb.conf in a text editor. Inspect the following commands, and revise their values as required. For example, you need to make sure that the following command is set to the name of your Windows Domain; my Windows Domain name is **grateful**.

```
workgroup = grateful
```



*As strange as it sounds, the smb.conf file uses the **workgroup** variable to identify the Workgroup or the Domain.*

Because you're working with a Domain-authentication database on a different computer, you'll need to cite that with the following command:

```
security = domain
```

Then you need to tell Samba how to look for the PDC machine. You can specify its NetBIOS name or IP address. If you don't know either, you can set it up to look via a broadcast message by substituting an asterisk (*). My Microsoft Windows PDC's NetBIOS name is **sugaree**.

```
password server = sugaree
```

To ensure that the users and groups from the Windows PDC do not interfere with those on your Linux computer, you should tell Samba how to configure Windows Domain users and groups on your computer. The following commands, which are beyond the current capabilities of the Red Hat Samba Configurator, add a range of User IDs and Group IDs outside the range of normal Linux users and groups:

```
# Set a new range of user and group IDs that do not interfere with the normal
ID range for Linux users and groups.
winbind uid = 5000-6000
winbind gid = 5000-6000
```

Because this computer is not a PDC, do *not* activate the following commands closely associated with a PDC:

```
# domain master
# domain logons
; logon script
; logon path
```

To deactivate a command, precede it with a comment character (# or ;).

For details of other commands in the smb.conf file, see Chapter 3, "Setting Up Your Server File System." As noted in that chapter, save your changes and then test the syntax with the **testparm** command.

Starting winbind

Before you start the **winbind** daemon, you need to add the administrator account from the Linux or Windows PDC computer to your Samba password database. On my network, the Windows Domain name is **grateful** and the Windows PDC has a NetBIOS name of **sugaree**. Substitute your computer names accordingly. I've done this on my Samba Domain member server with the following command:

```
# smbpasswd -j grateful -r sugaree -U administrator
```

If the PDC is located on a Linux computer, substitute *root* for *administrator*.



If this command does not work, you may have forgotten to set the WINS service on the PDC, or a firewall might be blocking traffic on the Microsoft or Linux computer. WINS communicates through TCP/IP port 1512, as discussed in Chapter 3, "Setting Up Your Server File System."

Now you can start the **winbind** service with the following command:

```
# /sbin/service winbind start
```

Now valid Domain users on any computer on the network can access the authorized shares from this Samba-based Domain member server. It's time to run a few commands to see if **winbind** is working properly. First, this command checks the status of **winbind**:

```
# wbinfo -p
```

You should get a message like "'ping' to winbindd succeeded"; otherwise, the **winbind** service may not have started correctly. Next, you can check to see that the computer account from this Domain member server is recognized on the PDC:

```
# wbinfo -t
```

You should see a message like "Secret is good." Now you can see if **winbind** is reading the list of Domain Users and Groups from the PDC with the following commands:

```
# wbinfo -u
# wbinfo -g
```

Finally, you can see how **winbind** puts it together with your /etc/passwd and /etc/group files with the following commands:

```
# getent passwd
# getent group
```

You should see a combination of the local authentication database file with the corresponding database from the PDC. For example, the **getent passwd** command lists the /etc/passwd file followed by a list of users from the PDC.



If you add another user, the next User ID number will come after those assigned to the Domain Users from the PDC. Based on the commands shown in this chapter, if there are four Domain Users, they get User IDs 5000, 5001, 5002, and 5003; the next user you create, locally or on the PDC, gets a default User ID of 5004.

Setting up a Samba PDC database

You can replace a Microsoft PDC with a Linux computer properly configured with Samba. If you're ready to make the conversion, you'll need to make some basic changes to the /etc/samba/smb.conf file, which I described in Chapter 3, "Setting Up Your Server File System." You'll also need accounts for each computer and user who will connect to your Domain.

To set up a Linux computer as a PDC configured with Samba, take the following four basic steps. Because I've already covered each topic in detail, I summarize the actions here.

1. Set up the accounts you need on the Linux computer. You can use the command **/usr/sbin/useradd *username***, or use the Red Hat User Manager. Linux user names are limited to 32 characters.



In Linux, case matters. Linux does not allow you to start user names with numbers. Linux also does not allow uppercase characters.

2. If you already have Microsoft Windows user names, you may want to keep them. If you have Linux user names and know their corresponding Microsoft user names, run the following command to match the two, and then, when prompted, enter the Microsoft user name's password. You can also use the Samba Server Configurator to set up the match as described in Chapter 3, "Setting Up Your Server File System."

```
# smbadduser linuxusername:microsoftusername
```



*Once you've set up a Samba password, you can change it with the **smbpasswd username** command.*

3. Configure accounts for each computer on your network that will connect to your Samba-enabled Linux PDC. As described earlier, you can use the **/usr/sbin/useradd** command to set up a computer account in /etc/passwd and then use the following command to add it to the Samba database.

```
smbpasswd -am computername
```



*You can administer a Linux PDC remotely. If you're on another Linux computer on the network, you can log in as user **mj** by using the **ssh mj@pdc_ computer** command. Because this is the secure shell, SSH communication is encrypted. If you're on a Microsoft Windows computer on the network, download the OpenSSH on Windows package from Network Simplicity at **www.networksimplicity.com/openssh** and then run the same **ssh** command.*

4. Recheck the main Samba configuration file. **Table 4** summarizes the key commands required to set up a PDC.

***Table 4**. Key commands in the smb.conf file to set up a Samba PDC.*

Command	Function
workgroup	Set this to the name of your Microsoft Windows Domain.
netbios name	Assign this to the NetBIOS name of your computer.
server string	Use this comment command to identify the PDC.
security	For a PDC, set this to **user**.
encrypt passwords	Set this to **yes** to match the encryption from Windows 95 OSR2 and later.
smb passwd file	Set this to the file name with Linux user names and Windows passwords.
unix password sync	When a Windows user changes a password, this command ensures that the password changes on the Linux computer.
local master	Supports election as the master browser on the local netwtork.
os level	Specifies the level required for election as the master browser for the Domain; set this to 63 or higher for a PDC with Windows 2000/XP workstations.
domain master	Supports election as the master browser for the Microsoft Windows Domain.
preferred master	Sets up a browser election when this Linux computer joins the Domain.
domain logons	Allows this PDC to manage Domain logins.
logon path	Specifies the directory for roving profiles from Windows NT/2000/XP computers.
wins support	Sets up a Windows Internet Name Service (WINS) server on the PDC.

Now you're ready to start a Linux PDC using Samba. If you have a new network, you can set up your Microsoft Windows workstations to log in to your new PDC. This topic is covered in Chapter 6, "Connecting Windows Workstations."

Logon scripts

Logon scripts are good for configuring basic commands for a workstation, such as mounting network drives. For example, let's look at a Microsoft Windows workstation, and a shared directory named project on a Linux member server computer named **nopaws**. The PDC is also installed on a Linux computer; you have set up a user named **nc** on the PDC's database. If you want to set up user **nc** to mount the shared project directory from a computer named **nopaws** on drive T:, create a Microsoft Windows text file named nc.bat in the /home/netlogon directory, and add the following line to it:

```
net use T: \\nopaws\project
```

This assumes that user **nc** has appropriate permissions to this share, as defined in smb.conf. I've set up the following commands in my smb.conf file:

```
logon script = %U.bat
```

and

```
[netlogon]
  comment = Network Logon Service
  path = /home/netlogon
  guest ok = no
  writable = yes
```

When a user logs in to the Domain, the PDC substitutes the name of the user for the **%U** variable. Thus, when user **nc** logs in to the Domain, the PDC uses smb.conf to look for the nc.bat file in the [netlogon] share, the /home/netlogon directory.

You'll need to save the desired commands, in Microsoft text format, in user-specific batch files. In this case, you'd save this script in /home/netlogon/nc.bat. The next time user **nc** logs in to the Domain from a Microsoft workstation, Samba sends nc.bat over the network to be run on that client computer.



You could set up a logon batch file with a Linux text editor, but Microsoft workstations need text files saved in Microsoft text format. I'll show you how to set this up in detail in Chapter 6, "Connecting Windows Workstations."

Roaming profiles

A roaming profile preserves a snapshot of a user's desktop, background, Start menu items, and more on a central server, usually a PDC. When the user who owns a roaming profile logs in to any workstation on that network, that user gets the same snapshot.

If you have roaming profiles on your network, the actual location of the profile varies depending on whether the computer is Windows 95/98/ME or Windows NT/2000/XP. The roaming profile for Windows 9x/ME clients is stored in that user's home directory on the Linux PDC, based on the commands described in Chapter 3, "Setting Up Your Server File System."

In contrast, the location of a roaming profile from a Windows NT/2000/XP computer depends on the **logon path** variable. Based on the value of the following variables, roaming profiles for Windows NT/2000/XP computers are stored in the /home/profiles/username directory.

```
logon path = /%L/Profiles/%U
path = /home/profiles
```

For example, if user **nc** wanted to log in to a Windows 2000 Professional computer, any roaming profiles that you configured are stored in the /home/profiles/nc directory.

For a detailed review of how to set up roaming profiles from Windows workstations on a Linux PDC, read Chapter 6, "Connecting Windows Workstations."

File and directory management

For those who are less familiar with Linux, it's important to understand the nature of files, directories, ownership, and permissions. Before showing how Linux and Windows permissions work together, I need to explain in detail how these concepts work in Linux.

In Linux, everything is represented by a file. Directories, drives, drivers, and links are just special versions of a file. Each file is associated with permissions for the user and group that owns the file.

Linux permissions

Take a look at permissions on your Linux computer. The best place to start is with your home directory. I can review permissions for my home directory by running the following command (to see the permissions on your home directory, substitute your user name for **mj**):

```
ls -l /home/mj
```

Here are three entries from my home directory on a Linux PDC:

```
drwxrwxr-x    3 mj    mj           4096 May 14 16:45 Start Menu
-rw-rw-r--    1 mj    mj         106528 May 14 19:24 USER.DAT
-rwxr-xr--    1 mj    project     46528 May 14 19:24 script
```

The permissions for this file are listed on the left side of each entry. It's important to analyze this in detail. The first letter indicates the type of file; the other letters are **r**, **w**, or **x**, as listed in **Table 5**.

***Table 5**. File permissions.*

Letter	Function
c	driver
d	directory
l	link to another file
r	read
w	write
x	execute
-	null character; when in the first position, it's a regular file

The remaining nine letters are associated with permissions for three entities: the owner of the file, the group that owns the file, and all other users on the system.

Each file identifies the user and group that owns that file. The user who owns each file is shown in the third column. The group that owns each file is shown in the fourth column. In my /home/mj directory, the owner of each file is user **mj**; the group that owns the file named script is **project**.

Consider the permissions for the script file shown above. The second, third, and fourth letters are **rwx**, which are associated with the owner of this file, **mj**. It means that user **mj** can read from, write to, and execute this script.

The next three letters are associated with the file permissions for the group that owns the file. Users in the **project** group can read and execute but can't write to the script file.

The final three letters define the file permissions for all other users on the system, who in this case can read only what's in the script file.

Default permissions

Whenever you create a file in Linux, that file gets a default set of permissions. In Red Hat Linux 9, those permissions are:

```
rw-rw-r--
```

In Linux, these letters are translated into numeric codes as defined in **Table 6**.

***Table 6**. File permission numeric values.*

Permission	Value	Description
r	4	r is assigned a value of 4
w	2	w is assigned a value of 2
x	1	x is assigned a value of 1
wx	3	set to the value of w + x = 3
rx	5	r + x = 5
rw	6	r + w = 6
rwx	7	r + w + x = 7

Take a look at the list of files from my home directory, /home/mj. In that directory, I can translate the permissions for the USER.DAT file to a numeric code of 664. The corresponding numeric code for the script file is 754. In other words, the user and group that owns USER.DAT can read and write that file; others on that computer can read the file. Others are able to read or copy from the file.

Now that you know these numbers, you can change the permissions associated with a file. For example, the following command keeps the script from being executable by anyone on that system:

```
# chmod 644 /home/mj/script
```

When you then run the **ls -l /home/mj** command again, you can see how the permissions have changed:

```
-rw-r--r--    1 mj    project     46528 May 14 19:24 script
```

As you can see, the **x** is now gone from the left side of the file listing, which means that the script is no longer executable by the owner of this file, **mj,** or any member of the **project** group.

Linux file ownership

Permissions define what the user, the group, and every other user on your network can do with a file. You can change the identity of the user and the group that owns that file. The **chown** command transfers the ownership of a file from one user to another. Consider the case of a script on a shared directory. Users **mj** and **nc** are both engineers working on the script. If I run the following command, user **nc** would get ownership of the script file:

```
# chown nc /home/engineer/script
```

This is shown by the following output:

```
# ls -l /home/engineer
-rw-r--r--    1 nc    project     46528 May 14 19:24 script
```

Let's say that someone decides to reassign responsibility of the script from the **project** to the **mfg** group. You could set this up with the following command:

```
# chgrp mfg /home/engineer/script
# ls -l /home/engineer
-rw-r--r--    1 nc    mfg         46528 May 14 19:24 script
```

Limited support for Access Control Lists

An Access Control List (ACL) is a Microsoft Windows concept for user and group rights and permissions. The details go beyond the ownership and permissions associated with a standard Linux file. ACLs are beyond the scope of this book.

Samba version 2.2.7, which is included with Red Hat Linux 9, contains greatly improved support for the Microsoft concept of ACLs. When you share files and directories from Microsoft Windows workstations, you can set up Domain users and Domain groups with the rights and permissions that you need—even if you've configured the PDC on a Linux computer. You'll see examples of how this works in Chapter 6, "Connecting Windows Workstations."

ACLs specify the rights and permissions that users and groups in a Domain have over specific files and directories. As of this writing, you can set up ACLs from Microsoft Windows workstations and Microsoft Domain member servers through a Red Hat Linux PDC. However, you can't set up ACLs directly from a Red Hat Linux PDC or Domain member server. I believe this situation will change soon with upcoming releases of Samba.

Linux groups

Microsoft Winodws servers include, by default, a series of groups, such as backup oeprators, power users, and so on. Linux does not include these standard groups. Nevertheless, you can set up directories that are readable only by users in a specific, private group.

Red Hat Linux's private groups

Red Hat Linux organizes standard groups differently from Microsoft Windows, and for that matter, differently from most other Linux distributions. Every user on a Red Hat Linux computer belongs to its own private group. For example, if you've created a user named **gecko**,

you'll also have a group named **gecko**. You can see how this works by looking at the /etc/passwd and /etc/group files. You'll find regular users at the end of each of these files.

Other Linux distributions organize all users as part of the same group, called **users**. All users on other Linux distributions normally have some level of access to each other's files.

In contrast, when you create a file in Red Hat Linux, you own the file. You are normally the only member of the group that owns the file. Other regular users on the computer normally don't get any more access to your files than guest users.

Creating a special group

Frequently, networks bring together groups of users who need access to a common set of files on a single directory. It's not hard to set this up in Linux. You learned at the end of Chapter 3 how to set up a common directory on a Linux PDC or Domain member server for a group of users. In this section, I'll put these skills together in an example.

Assume you want to set up a private directory for the engineers in an airplane seat factory: Gabe, Waymon, Padma, and Diane. The lead engineer is Gabe. To make it work, follow these steps:

1. Log in to the Linux PDC. Open the Red Hat User Manager by clicking Main Menu | System Settings | Users and Groups. Click Add User to open the Create New User window shown in Figure 2. Create an account for each seat engineer. Alternatively, you can use the following shell commands to create user accounts for the seat engineers:

```
# /usr/sbin/useradd gabe
# /usr/sbin/useradd waymon
# /usr/sbin/useradd padma
# /usr/sbin/useradd diane
```

2. If your seat engineers want to log in directly at a Linux workstation, create a password for each user on that workstation with the **passwd *username*** command. If they're just using Microsoft Windows workstations, you don't need to create a Linux password for each user. Shortly, I'll show you how to set up Samba passwords, which these users would enter to log in to a Windows workstation on the Domain with the Linux PDC.



As of this writing, Samba does not support logins to Linux workstations with accounts that reside only on a Windows or Linux PDC.

3. Set up a group for the seat engineers. In the Red Hat User Manager, click Add Group. In the Create New Group window shown in **Figure 5**, set up an appropriate group name. It's also best to set up a Group ID manually outside the range of normal User IDs, as explained earlier.

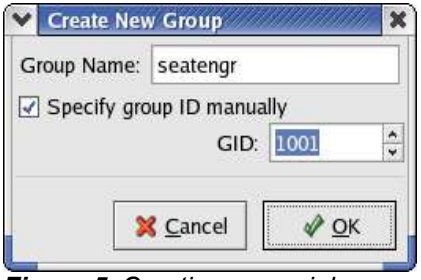


***Figure 5**. Creating a special group.*

4. Create a special directory for the seat engineers. Assign user ownership to the lead engineer (Gabe), and group ownership to the seat engineering group. Allow the user and group full permissions on this directory.

5. Configure the directory so its files are owned by the special group. Keep others from reading from this directory (the Linux administrative user, root, still has access). It's easiest to do this from the command-line interface:

```
$ su
Password:
# mkdir /home/seatengr
# chown gabe /home/seatengr
# chgrp seatengr /home/seatengr
# chmod 770 /home/seatengr
# chmod g+s /home/seatengr
```



*The **chmod g+s /home/seatengr** command sets the Group ID bit on the /home/seatengr directory. Any file copied to or created in this directory inherits the group owner of this directory. In other words, all members of the **seatengr** group have full default access to all files in the /home/seatengr directory.*

6. In the Red Hat User Manager, click the Groups tab. Highlight your new **seatengr** group and click Properties. In the Group Properties window, click the Group Users tab shown in **Figure 6**. Select your seat engineer users from the list, and then click OK. Now you have a special directory, /home/seatengr, where access is limited to your seat engineers.

7. Set up each user for Windows logins. You can do this in two ways. If you want to use the Red Hat Samba Configurator, click Main Menu | System Settings | Server Settings | Samba Server. In the Samba Server Configuration utility, click Preferences | Samba Users. In the Samba Users window, click Add User to create Samba database entries for each seat engineer. **Figure 7** illustrates that Waymon's Windows user name is **waymonww**.



***Figure 6**. Selecting users for a group.*

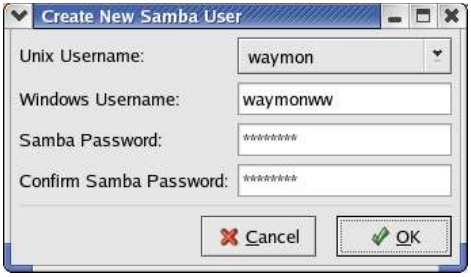


***Figure 7**. Creating a Samba user for Windows Domain logins.*

8. Alternatively, you can do this from the command line with the **smbadduser** command. For the case shown in Figure 7, run the following command:

```
# smbadduser waymon:waymonww
```

You're prompted for the password that Waymon would use when logging in from a Microsoft workstation.

9. Add a share to the Samba configuration file. Limit access to the members of the seat engineering group. Based on the discussion in Chapter 3, you can do this by adding the following commands to the end of the Samba configuration file:

```
[seatengr]
   comment = Seat Engineers Directory
     path = /home/seatengr
   valid users = @seatengr
   public = no
   writable = yes
   printable = no
   create mask = 0770
```

10. Check the syntax of your changes with the **testparm** command. Address any problems cited by **testparm**. Restart the Samba daemon with the following command:

```
# /etc/rc.d/init.d/smb restart
```

Now your Domain is ready, configured for a group of seat engineers.

Quotas

Quotas keep users from being selfish, disorganized, or sloppy. They can limit the space or number of files taken by any single user or group. Users who take up too much space can crowd out other users, the space you have for data, or even the space you need to boot a Linux computer. If you configure home directories on a Linux computer and don't have unlimited disk space, you'll want to set up quotas.

You can limit the damage caused by any particular user by using partitions, as discussed in Chapter 1, "Basic Linux Installation." Even if you've set up a separate partition for the /home directory, you'll still have the potential problem of users who crowd out others. Unfortunately, there is no GUI installed with Red Hat Linux 9 that can help you configure quotas.

The Red Hat Linux **quota** RPM is installed by default. So all you need to do to set up user and group quotas is to take the following steps:

1. Set up the main Linux partition configuration file (/etc/fstab) to start quotas the next time you mount the desired directory.
2. Remount the target directory so you can enable quotas without rebooting Linux.
3. Set up the quota configuration file for users or groups.
4. Create or modify the quota file for the mounted directory.
5. Configure the quotas you've created for a set of users or groups.
6. Modify the grace period as desired.
7. Activate quotas with the **quotaon** command.

Once you've configured quotas, you can use the **edquota** and **repquota** commands to check up on your users. In the following sections, I explain each step of the process in detail.

The amount of data you assign to each user depends on the space you have available on your hard drive, and what your users do in your business. If users are just creating office-suite files, they need considerably less space than if they are creating 3-D solid Computer-Aided Design models of airplane galleys.

The boot process

When you start Linux, it loads the kernel, basic startup files, and more. During this process, Linux mounts directories such as /boot and /home on specific partitions. The file /etc/fstab defines the way these directories are mounted.



Whenever you edit a key configuration file such as /etc/fstab, back it up first. You could save it on a floppy disk or to the /root directory. If you have problems booting in the future, restore your system by using the rescue disk techniques described in Chapter 8, "Administration and Management."

Take a typical line from this file:

```
LABEL=/home          /home                ext3        defaults  1 2
```

This line defines how the /home directory is mounted on this computer—to the third extended filesystem (ext3). Because this is not a book about the inner workings of Linux, I'll focus on the fourth column, which defines how the directory is mounted.



If you did not set up the /home directory on a separate partition during the installation process (see Chapter 1), you won't see it in /etc/fstab.

Generally, you won't need to change what's been configured in /etc/fstab. To configure quotas, all you need to do is add **usrquota** or **grpquota** to the fourth column depending on whether you want to configure user or group quotas on that directory.

For example, if you're setting up user and group quotas on the /home directory in /etc/fstab, change the applicable line to:

```
LABEL=/home          /home          ext3     defaults,usrquota,grpquota  1 2
```

You may need to remove some spaces to make sure all the information associated with the /home directory remains on one line. You don't need to reboot to put user and group quotas into effect. Once you've edited and saved your /etc/fstab file, just remount the directory. For the example shown with the home directory, the following command should work:

```
# mount -o remount /home
```

You can verify the result. When you run the **mount** command, you'll see a list of mounted directories. For the aforementioned command, you should see something similar to the following as part of the output:

```
/dev/sdb2 on /home type ext3 (rw,usrquota,grpquota)
```

This particular result shows the /home directory mounted on the second primary partition on the second SCSI hard drive. Your result depends on where you've configured /home. For more information on this naming convention, see Chapter 1, "Basic Linux Installation."

Configuring quota configuration files

Before you can configure a quota on Linux, you'll need to set up a quota configuration file. It's a simple process. For example, to set up the user quota configuration file on the /home directory, run the following commands:

```
# touch /home/aquota.user
# chmod 600 /home/aquota.user
```

The **touch** command creates the aquota.user file in the /home directory with zero bytes; this **chmod** command makes it readable and writable only by the root (administrative) user. Others who don't have access to the root user password can't read or change this file. Similarly, to set up the group quota configuration file on /home, run these commands:

```
# touch /home/aquota.group
# chmod 600 /home/aquota.group
```



If the mounted directory is different, such as /usr or /, create the quota file in that directory.

Now you can configure each of these files. The following commands scan (**-a**), return verbose output (**-v**), and remount the /home directory. Naturally, the first command performs this task for user quotas (**-u**), and the second command performs this task for group quotas (**-g**). Finally, it does all of this without remounting (**-m**) the subject directory.

```
# /sbin/quotacheck -avum
# /sbin/quotacheck -avgm
```

Configuring quotas for a user

The next step is to set up quotas for an individual user. For example, I can inspect the current quotas for user **jkp** with the following command:

```
# /usr/sbin/edquota -u jkp
```

This opens the current quota configuration file in the vi text editor, which I describe in Chapter 8, "Administration and Management." I'll give you step-by-step instructions shortly. But first, take a look at the configuration file on my computer:

```
Disk quotas for user jkp (uid 506)
   Filesystem         blocks     soft     hard     inodes     soft     hard
   /dev/sdb2              68        0        0          9        0        0
```

There are seven columns in this file. In this case, the filesystem defines the partition where I've mounted the /home directory; the next three columns define limits in KB; the final three columns define limits in inodes, which in this case corresponds to the number of files owned by user **jkp** in the /home directory. In other words, user **jkp** has nine files in /home; these occupy 68KB of space. Each group also includes soft and hard limits. The "0" in the other columns means that there are no limits.



With the latest versions of Linux, there are few practical limits on the number of files, so quota limits for files are rare. (The current limit is encoded into the Linux kernel in the /proc/sys/fs/file-max file.)

A soft limit defines the maximum amount of space or number of files associated with this user. If there's a grace period, the user will have to reduce the number of files and/or the space he or she uses. The default grace period is seven days; I'll describe how you can change this shortly.

A hard limit applies if you've set a grace period. After the grace period, Linux locks the account unless the user reduces the space and number of files below these limits.

I've summarized each column in **Table 7**.

***Table 7**. User quotas.*

Column	Function
1	Filesystem, which defines the partition with the quota. The quotas you define apply to any directory mounted on that partition.
2	Blocks define the space currently taken by the user in the specified filesystem, which corresponds to kilobytes. For the example shown above, user **jkp** currently has 68KB of files in the /home directory.
3	The first soft limit defines the space, in KB, that the user is allowed to store in the /home directory.
4	The first hard limit defines the space, in KB, that the user can store in the /home directory—after the grace period expires.
5	The inodes column defines the number of files owned by the user in the /home directory.
6	The second soft limit defines the number of files that a user can store in the /home directory.
7	The second hard limit defines the number that a user can store in the /home directory after the grace period expires.

Now let's start editing quotas. For this example, I'm going to set a hard limit on user **jkp** of 100MB; I'll also include a 10 percent margin. Because inode quotas are rarely needed, I won't set them for this user. To edit the space quota for user **jkp**, take the following steps:

1. Start a command-line interface. Open the quota file for user **jkp** with the **/usr/sbin/edquota -u jkp** command.
2. I've arbitrarily set a 10 percent margin for user quotas. You can set the margin of your choice. With a 10 percent margin, the soft limit will be 110MB. Because the information is in KB, you'll want to enter 110000 under the first "soft" column. I'm assuming you've opened the quota file in the vi editor.
3. Use your cursor to move to the desired location in the file, and then type the **i** command (which enters "insert mode").
4. In the version of vi normally installed with Red Hat Linux, you can then use the backspace, delete, arrow, and other keyboard keys to make the desired change.
5. When you're finished, press the Esc key (which returns to "command mode").

6. The desired limit is 100MB. In the first "hard" column, enter 100000, which is 100MB in KB. Use your cursor to move to the desired location in the file, and type the **i** command again. Repeat the process to enter the soft limit. When you're done, the file should look similar to this:

```
Disk quotas for user jkp (uid 506)
  Filesystem    blocks    soft      hard     inodes     soft     hard
  /dev/sdb2        68     110000    100000     9          0        0
```

7. Save and write your changes with the **:wq** command. This writes the file, closes the vi editor, and returns you to the command-line interface.



The vi editor can be intimidating to anyone accustomed to GUI editors. Unfortunately, it isn't currently possible to open a Linux quota file in a GUI editor. If you're not familiar with Linux text editors, you may want to read about vi in Chapter 8, "Administration and Management."

You can confirm your changes to user **jkp** with the following command:

```
# quota jkp
```

Configuring quotas for a group

Now you can repeat the process to configure quotas for a group. The basic commands are essentially the same, with minor variations. You've already set the basic quota configuration file, /home/aquota.group. If you want to set a quota for the seat engineering group described earlier, take the following steps:

1. Run the following command to open the group quota file, in this case the **seatengr** group.

```
# /usr/sbin/edquota -g seatengr
```

2. The vi editor opens by default. When it opens, modify the hard and soft limits as desired. As described in the previous section, use the **i** command to enter "insert mode," where you can use your keyboard to insert and delete as needed. Press the Esc key when done.
3. Run the **:wq** command to write the file and close the vi editor.
4. Confirm your changes to the group quota with the following command:

```
# quota -g seatengr
```

Setting a grace period

Based on the example in this chapter, the default grace period is seven days. The soft limit is 110MB and the hard limit is 100MB. If I store more than 100MB in the /home directory, the clock on the grace period starts. I need to get under the 100MB hard limit before the grace period ends. To open the file that sets the grace period, use the following command:

```
# /usr/sbin/edquota -t
```

The grace period file is shown in **Figure 8**. As you can see, it's a simple file. If you want to change the grace period, you should note that there is no space between "7" and "days" under the grace period columns. If you add a space, quotas will no longer work.

```
Grace period before enforcing soft limits for users:
Time units may be: days, hours, minutes, or seconds
  Filesystem             Block grace period     Inode grace period
  /dev/sdb2                     7days                  7days
~
```

***Figure 8**. Grace periods.*

The Filesystem is the partition associated with the /home directory. You can verify it with the following command:

```
# e2label /dev/sdb2
/home
```

The Block grace period is associated with the space limits for each user. If you want to change the grace period, take the following steps:

1. Move the cursor to the number.
2. Enter the **cw** command. This should delete "7days".
3. Enter the period of your choice, such as **5days**. Remember, don't add a space between the number and period.
4. Press the Esc key.
5. Enter the **:wq** command to write your changes to the grace period.



*If you want to exit from the vi editor without saving, use the **q!** command.*

Activating quotas

Quotas may not be much help unless you can apply them to a group of users. Fortunately, this is easy with the **edquota** command. For example, the following command applies the same quota that you configured for user **jkp** to the other users in this list:

```
# edquota -up jkp gabe mj padma waymon diane jc nc vf wg
```

The **-u** switch specifies users; the **-p** switch sets the first user as the "prototype" quota to apply to the other users on the list. You can repeat the same process for groups; the following command applies the quota from the **seatengr** group to the other groups on the list:

```
# /usr/sbin/edquota -gp seatengr maryfred canmex
```

Once the file is configured, it's easy to enable quotas. Just run the following command:

```
# /sbin/quotaon
```

And that's it. The next time you reboot Linux, it uses the settings you changed in /etc/fstab to find the quota files that you created during this process, and then it activates them automatically.

Conclusion

In this chapter, you learned to create and configure the users and groups that you need to set up a Linux computer as a Domain member server or a PDC. There are three different types of user accounts on a Linux computer with Samba: regular Linux users, Samba users for logins from Microsoft Windows workstations, and computer accounts to accommodate connections from those same Windows workstations.

If you're keeping a Microsoft Windows computer as a PDC, you can set up the **winbind** daemon on Linux-based Domain member servers. It'll then use the user name and password database from the PDC to share files and printers from the Linux Domain member server.

If you're configuring a Linux computer as a PDC, the right settings in the Samba configuration file, smb.conf, support roaming profiles. The effect is slightly different depending on whether you're connecting from Windows 9x/ME or Windows NT/2000/XP workstations.

To understand the interactions between Linux and Microsoft Windows, you need to understand the nature of ownership and permissions files and directories in Linux. Fortunately, even a Linux-based PDC supports the use of ACLs from Microsoft workstations and member servers. However, you can't set ACLs directly on current Samba servers that are based on Red Hat Linux 9.

While Linux doesn't have the diversity of default groups available on Microsoft Windows servers, you can configure special groups with rights and permissions limited to member users. In this chapter, I showed you how to create these groups directly, or through the Red Hat User Manager.

Finally, you can regulate the resources taken by users and groups by using quotas. While there is no Red Hat GUI tool that you can use to configure quotas, the commands you need to set up quotas are straightforward.

> Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 5
Connecting Linux Workstations

The same Red Hat Linux that is making a name as a server can also be configured as a powerful desktop and workstation computer. Naturally, you can connect a Linux computer to a network of Microsoft Windows and Linux computers. In fact, configuring Linux as a workstation is a natural option for Windows users who want to try Linux for the first time.

You don't need to install every Samba package to connect a Linux computer as a workstation. You just want the basic Samba client packages to connect to the shared directories of your choice. But if you also want to share directories or printers from your Linux workstation, that workstation will require the Samba server package.

One drawback to running a Linux workstation on a Microsoft Windows Domain is that you can't log in to the local workstation using the Windows authentication database from a Microsoft Windows or Linux PDC. (You could use the Network Information System or the Lightweight Directory Assistance Protocol to set up a single authentication database; these technologies are beyond the scope of this book.)

However, once you've logged in to a Linux workstation, you can connect to shared directories on a Windows-based network. When you make the connection, you'll enter a user name and password for the Domain or the member server. If the workstation is part of a Domain, the user name and password can also come from any user that's a member of the Domain.

You can set up the Domain user name and password in Linux startup files. This way, users who log in to a Linux workstation are automatically connected to shared directories and printers on the Domain. In this chapter, I'll show you how to set this up for individual users.



A number of books on Samba help you configure automatic connections through the /etc/fstab configuration file. I avoid this file in this book, because you can't use it over a network to customize mounted directories by user.

The steps you take to connect to a shared directory or printer vary slightly. At the end of this chapter, I'll show you how to connect your Linux workstation to a PDC or a Domain member server on a Microsoft Windows-based Domain. The steps vary slightly depending on whether you're connecting to a share on a Windows or a Linux server.

If you have problems connecting to a Samba server or PDC, there are a number of log files that may help. With the standard smb.conf file, there are individual log files for every computer that connects. These are stored in the /var/log/samba directory. I describe these files in some detail in Chapter 6, "Connecting Windows Workstations." The messages from these log files apply equally well to Linux workstations.

Many of the commands described in this chapter require administrative or root user access. If you've logged in as a regular user (as you should), you can get root user access with the following command:

```
$ su
Password:
#
```

Configuring the workstation

Just as you can configure a share from a Microsoft workstation on a Domain, you can configure a share from a Linux workstation on a Domain. However, to share directories or printers from a Linux workstation, you also need to install the Samba server software.



When I talk about a Microsoft Windows-based network, I'm talking about the standard networking scheme associated with Microsoft operating systems.

Samba client packages

All you need to connect to shared directories on a Microsoft Windows-based network are the **samba-client** and **samba-common** RPM packages. If you followed the installation instructions in Chapter 2, you've installed the System Tools package group, which includes these RPMs by default. You can make sure these packages are installed by using the following commands:

```
$ rpm -q samba-client
$ rpm -q samba-common
```

If these packages are installed, you'll see output similar to the following:

```
samba-common-2.2.7a-8.9.0
samba-client-2.2.7a-8.9.0
```

Alternatively, you may see a message like "package samba-client is not installed." In that case, you can install these packages from the first Red Hat Linux 9 CD. Insert it into your drive as the root user and run the following commands from the command-line interface:

```
# mount /mnt/cdrom
# rpm -Uvh /mnt/cdrom/RedHat/RPMS/samba-common*
# rpm -Uvh /mnt/cdrom/RedHat/RPMS/samba-client*
```

Usually, all Microsoft Windows computers on a network can act as servers. Even Windows 9x/ME computers can share directories and printers on a Domain. If you want your Linux workstation to share directories or printers, you'll also need to install the **samba-2*** RPM server package, which is available in Red Hat Linux 9 from the same installation CD. For more information on the Samba server and sharing directories, see Chapter 3, "Setting Up Your Server File System," and Chapter 4, "Setting Up Your File Server's Users."

However, if you don't plan to share directories or printers from a Linux workstation, don't install the Samba server. Installing software that you don't use can be a security risk.

Connecting to a Domain

With a Linux client computer, you can connect to a Domain governed by a Windows or a Linux PDC. If you also want to share from the Linux client, you'll need the additional Samba server packages described in Chapter 3, and will need to set up your computer as a Domain member server as described in Chapter 4. Once the required packages are installed, there are several things you need to do:

- Make sure the Samba daemon is running by using the following command:

```
/sbin/service smb status
```

- Make sure the Winbind daemon is running by using the following command:

```
/sbin/service winbind status
```



If you have a problem with a daemon, restart it. For example, the following command stops and starts the Winbind daemon: ***/sbin/service winbind restart****. If you get a "subsys locked" error message, try joining your workstation to the Domain first.*

- Join your workstation to the Domain. The following command joins the local workstation to the GRATEFUL Domain, with a PDC on a Linux computer named **cosmicc** (if the PDC is on a Windows computer, substitute *administrator* for *root*):

```
# smbpasswd -j GRATEFUL -r cosmicc -U root
```

- Check your local Winbind connection to the PDC by using the following commands:
 - **wbinfo -p** checks the connection to the PDC.
 - **wbinfo -t** tests the workstation account on the PDC; if it is, you'll see a "secret is good" message.
 - **wbinfo -u** lists the users from the PDC.
 - **wbinfo -g** lists the groups from the PDC.

If you have trouble with one of these commands, you may need to restart the Samba or Winbind daemons. Alternatively, there may be a problem with the Samba configuration file as described in Chapter 4, "Setting Up Your File Server's Users." Typical problems are related to the **password server**, **wins client**, **winbind uid**, and **winbind gid** variables.

After you've verified each of these items, your computer is ready to be a client that can share directories and printers on the Domain.

Finding shared directories

It's not difficult to find a list of shared directories on a Linux workstation. As an administrator, you should know how to run this at the command-line interface. With this skill, you can set up

Linux versions of the login batch file commonly used to connect Microsoft Windows workstations to shared directories.

The command that allows you to browse shared directories is **smbclient**. The following example from a Linux workstation lists the names of shared directories and printers on a Microsoft Windows XP Professional computer named **allaccess**:

```
# smbclient -L \\allaccess -U michael
```

The basic **smbclient** command communicates with any server on a Microsoft Windows network. That includes any Linux computer running a Samba server, and any Microsoft Windows computer with shared directories or printers on a network.

The **-L** switch directs **smbclient** to look for the shares from a specific computer on a network. In this case, that computer is named **allaccess**. The **-U michael** at the end of the command specifies a user name on the **allaccess** computer, or a user name on the Domain. You're then prompted for that user's password.



*You can also include the password with the user name, in the following format: **-U michael%password**. However, this exposes your password to anyone who might be looking over your shoulder.*

If **allaccess** is a computer on a Domain, the aforementioned **smbclient** command searches for *michael* in the PDC authentication database. Once you've entered the appropriate password, you'll get a full list of shared directories accessible to that user. The output from my computer is shown in **Figure 1**.

```
[mj@Delilah mj]$ smbclient -L \\allaccess -U michael
added interface ip=172.22.30.38 bcast=172.22.30.255 nmask=255.255.255.0
Password:
Domain=[DARKSTAR] OS=[Windows 5.1] Server=[Windows 2000 LAN Manager]

        Sharename      Type      Comment
        ---------      ----      -------
        IPC$           IPC       Remote IPC
        D$             Disk      Default share
        print$         Disk      Printer Drivers
        Hentzen        Disk      Linux Transfer for Windows
        ftproot        Disk
        HPLaserJ       Printer   Comment Test
        Downloads      Disk
        RedHat         Disk
        ADMIN$         Disk      Remote Admin
        C$             Disk      Default share

        Server               Comment
        ---------            -------
        ALLACCESS            LaptopWin

        Workgroup            Master
        ---------            -------
        DARKSTAR             ALLACCESS
[mj@Delilah mj]$
```

***Figure 1**. Finding a list of shared directories and printers.*

Figure 1 illustrates the current list of shared directories and printers. Each item includes a share name and a comment that you specified when you created the share. **Figure 2** shows how this works with the Hentzen share from the **allaccess** computer.



***Figure 2**. Configuring the Hentzen share from a Windows directory.*

Mounting directories

Once you learn to mount a directory from the command-line interface, you can set up batch login files for each user. These batch files run the commands that automatically mount those directories the next time each user logs in to Linux.

As long as you have downloaded the Samba client packages described at the beginning of this chapter, you can mount shared Windows network directories by using the **smbmount** command. Of course, you need to specify the name of the computer, the share, and normally the user name and password.

Consider an example based on Chapter 4, where you set up a shared directory for a group of airplane seat engineers. On my Linux Domain member server, this shared directory is located at /home/seatengr on the **nopaws** computer in the grateful domain.

This is one case where it may be faster to use the Linux GUI, as customized by Red Hat. In the GUI, click Main Menu | Network Servers. This opens the Linux Nautilus browser, which views connections to computers on a Microsoft Windows network. If your Linux computer recognizes NetBIOS broadcasts, it should be able to view the Workgroups and Domains on your network, as shown in **Figure 3**.



*You can view a Windows network from any Nautilus browser by typing **smb:///** in the Location text box. This is a Uniform Resource Identifier (URI), which locates all Microsoft Windows and Linux computers with a Samba server on your network.*

Naturally, when you double-click the GRATEFUL icon, you'll see a list of servers and workstations in the GRATEFUL Domain. You'll see all Windows and Linux computers that are configured to share files and printers on the Domain. On my Domain, my PDC is named **cosmicc**, and my Linux member server is named **nopaws**.



***Figure 3**. Viewing a Domain in the Linux GUI.*

When I double-click the **nopaws** computer, I'm prompted for a user name and password, as shown in **Figure 4**. I can enter any Windows user name on the PDC or member server. You'll remember from Chapter 4 that Linux computers store a database of Windows user names in the /etc/samba/smbusers file, which is activated with the following command in the smb.conf file:

```
username map = /etc/samba/smbusers
```

I've entered the Windows user name **waymonww**. On the PDC, I've associated **waymonww** with the Linux user name **waymon** in the /etc/samba/smbusers file. I can do this with the **smbadduser** command described in Chapter 3, "Setting Up Your Server File System."

If the PDC or member server recognizes your user name and password, Nautilus then navigates to the shared directories. For the **nopaws** computer, that includes the shared directories as defined in its local smb.conf configuration file. These shares are shown in **Figure 5**. As you can see, this includes the shares from the end of the smb.conf file, as well as the basic home directory share. In this case, because the Linux user name is **waymon**, the share is from the /home/waymon directory.



***Figure 4**. Connecting with a user name and password.*

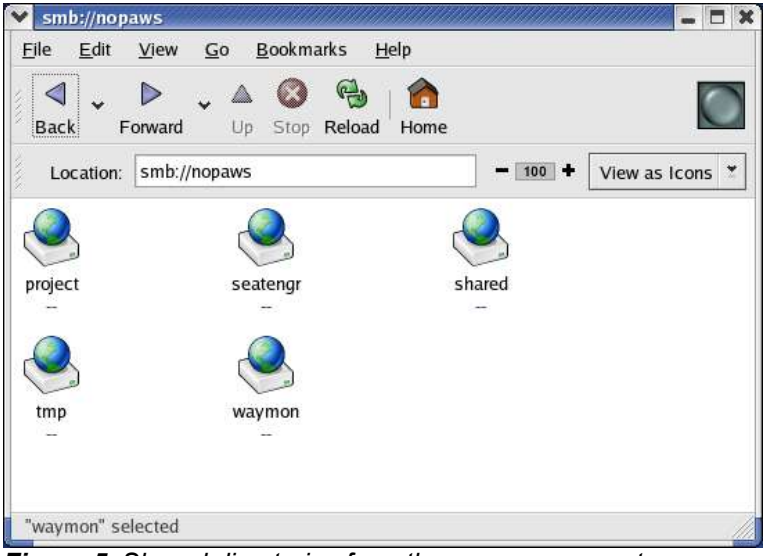


***Figure 5**. Shared directories from the **nopaws** computer.*



If you connect as a Domain user, you'll see a share named after that user in Nautilus—even if that user does not have a home directory on the target server.

Assuming the PDC or member server verifies your credentials, Nautilus takes you to a graphical view of the shared directories in that domain. I illustrate how this works in **Figure 6**, which is a graphical list of files from the [seatengr] share on the **nopaws** computer.

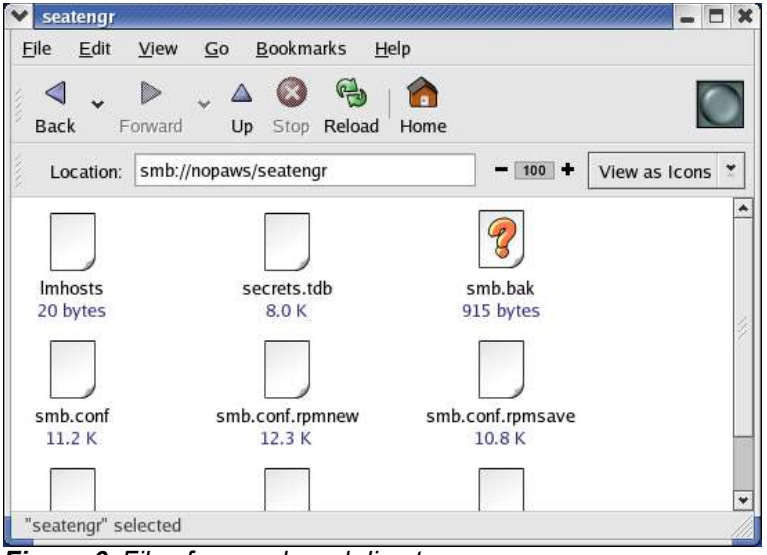


***Figure 6**. Files from a shared directory.*

You can now manage files from this directory graphically. For example, you can delete files by dragging them to the Trash icon on the desktop. You can copy files to directories open in other Nautilus windows. If Nautilus recognizes the extension, double-clicking the file opens it in an appropriate application such as the GNOME text editor.

But as an administrator, you'll want to set up your users with text commands in appropriate login batch files. First, let's test some commands that you could use. For example, if I wanted to mount the /home/seatengr directory from **nopaws**' [seatengr] share, I could run the following command:

```
# mount -o username=waymonww //nopaws/seatengr /home/seatengr
password:
```

This command mounts the files from the [seatengr] share on the **nopaws** computer on the local /home/seatengr directory. I can now list, edit, and manipulate the files from this share as if it were on the local computer. If you like, you can detach from the share (unmount the same directory) with the following command:

```
# umount /home/seatengr
```



*In Linux, the **umount** command unmounts directories (this is not a typo).*

There is one drawback to the **mount** and **umount** commands: You can run them only as the Linux administrative user, root. But most users won't log in as root. There are related commands that you can enable for regular users. To do so, take the following steps:

1. As the root user, run the following commands, which allow all regular users to mount and unmount directories shared over a Microsoft Windows network:

   ```
   # chmod u+s /usr/bin/smbmnt
   # chmod u+s /usr/bin/smbumount
   ```

2. Now you can mount and unmount shared directories as a regular user by using the **smbmount** and **smbumount** commands. As regular user **mj**, I've tested it as follows:

   ```
   $ smbmount //nopaws/seatengr /home/mj/seatengr -o username=waymonww
   passwd:
   $ ls /home/mj/seatengr
   $ smbumount /home/mj/seatengr
   ```

3. Now that you've seen how the **smbmount** and **smbumount** commands work for a regular user, you can set up each user's startup files to mount shared directories automatically.

Linux login batch files

On a Microsoft Windows workstation, you can set up *username*.bat or login.bat files to connect to shared directories automatically. You can set up the same functionality for users on Linux workstations.

By default, when a user logs in to the Linux workstation, Linux runs startup files for that user. The name of the startup file depends on whether you're logging in to the GUI or the command-line interface.

GUI login batch setup

You can set up commands that Linux runs when you log in to the GUI. From the Linux GNOME desktop, click Main Menu | Preferences | More Preferences | Sessions. This opens the Sessions dialog, which starts the Sessions manager. Click the Startup Programs tab as shown in **Figure 7**.

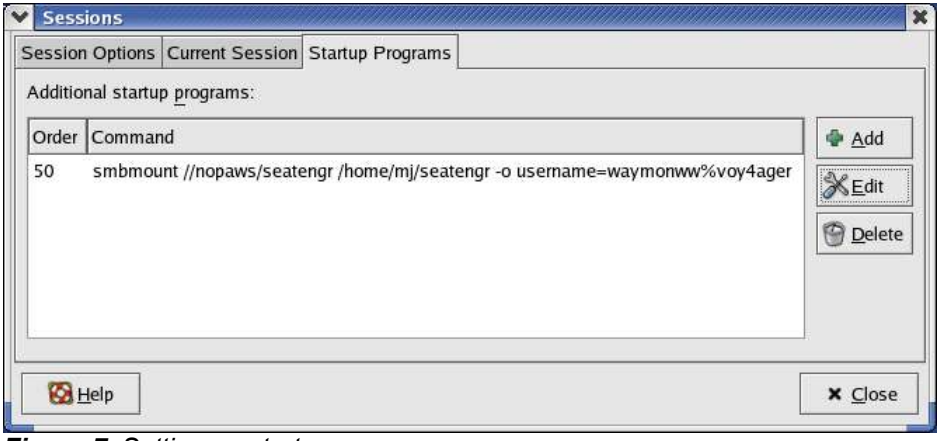


***Figure 7**. Setting up startup programs.*

If you want to add a startup program, click Add. In the Add Startup Program dialog, enter the command of your choice in the Startup command text box. As you can see from Figure 7, I've set it up with the **smbmount** command described earlier. I've also added the password (voy4ager) to the command. If you're configuring multiple startup commands, you can set up the command that runs first by giving it a higher order number.



The GUI Sessions manager won't mount a directory unless you've added the password. You can do this with the "%" character. In the example shown in Figure 7, waymon's password is voy4ager.

Any commands that you set up with the Sessions manager are saved in each user's home directory. For example, when I ran the Sessions manager as user **mj**, the command I added was saved in the /home/mj/.gnome2 directory, in the session-manual file.



In Linux, files and directories with a period (.) in front are hidden. Thus, the .gnome2 directory is normally hidden from view.

You can browse hidden files and directories from the command-line interface by using the following command:

```
# ls -a
```

If you want to open a file in a hidden directory with the GNOME text editor, gedit, follow these steps:

1. Click Main Menu | Accessories | Text Editor.
2. In gedit, click Open.
3. Type **.gnome2** in the Selection text box and press Enter; this gives you a view of the files in the hidden /home/mj/.gnome2 directory as shown in **Figure 8**.

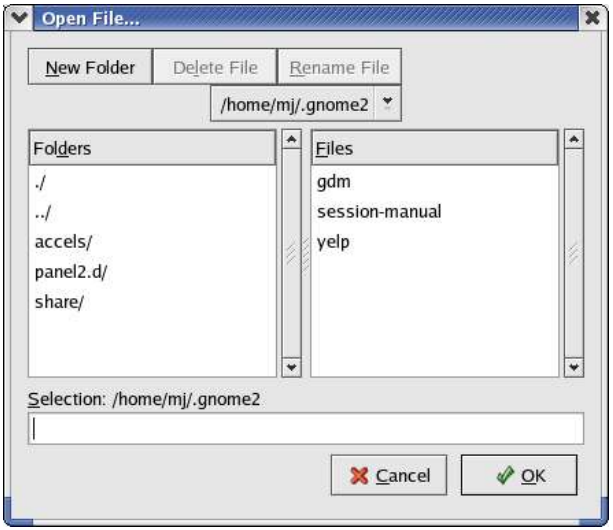


***Figure 8**. Opening a hidden file in the GNOME editor.*

4. Open the file of your choice. To open the GNOME startup script, highlight the session-manual file and click OK. Now you can add the batch commands of your choice. The next time this user logs in to the GUI, GNOME runs the commands in this file.

Text login batch setup

If you log in to a Linux workstation in text mode, the first thing you see after you start your Linux computer should be similar to the following:

```
Red Hat Linux release 9 (Shrike)
Kernel 2.4.20-9 on an i686

Delilah login:
```

Log in locally, and then run the following command to view all of the files in your home directory:

```
# ls -a
```

The key startup file in a Red Hat Linux home directory is .bash_profile. When you log in to a computer at a text console, Linux starts by default in the bash shell, which is a command line associated with a group of commands similar to MS-DOS. The .bash_profile file is where Red Hat Linux administrators set up custom logon scripts for each user.

You can open this file in the text editor of your choice. Because this book is directed at Microsoft Windows users, I'm assuming that you'll want to open it in a GUI. To enter the GUI in text mode, type the following command:

```
# startx
```

Once the GUI is open, you can open .bash_profile in the GNOME editor described earlier. Click Main Menu | Accessories | Text Editor. Click Open, and then type the name of the .bash_profile file as shown in **Figure 9**. Don't forget the period (.) at the start of the file.

You can add the command of your choice in the gedit editor. In **Figure 10**, I've added the same command as in the previous section, without the password.

Once you've saved this file, the next time you log in as this user, Linux runs the customized **smbmount** command automatically. The next time you log in at text mode, you'll be prompted for two passwords: The first password is for the user, and the second is to verify a connection to the shared directory:

```
Red Hat Linux release 9 (Shrike)
Kernel 2.4.20-9 on an i686

Delilah login: padma
Password:
Last login: Wed Jun 4 08:36:32 on tty1
Password:
```

That may be too much to handle. If you don't have a problem with the potential security risks, you can include the password in the .bash_profile script; just add the password (in this case, voy4ager) to the end of the command as shown:

```
smbmount //nopaws/seatengr /home/$USER/seatengr -o username=waymonww%voy4ager
```

If you enter the password in **padma**'s .bash_profile file, she won't have to enter that second password when she logs in to the text console.

If you have a lot of users, it's rather inefficient to edit every user's .bash_profile file. You can add the same command to the .bash_profile file for a number of different users. For example, the following command sets up the same mount in the .bash_profile file of user **padma**'s home directory:

```
# echo 'smbmount //nopaws/seatengr /home/$USER/seatengr -o username=waymonww' \
>> /home/padma/.bash_profile
```



In the previous command, the backslash (\) at the end of the first line is a continuation character; it tells Linux to read both lines together as one long command.

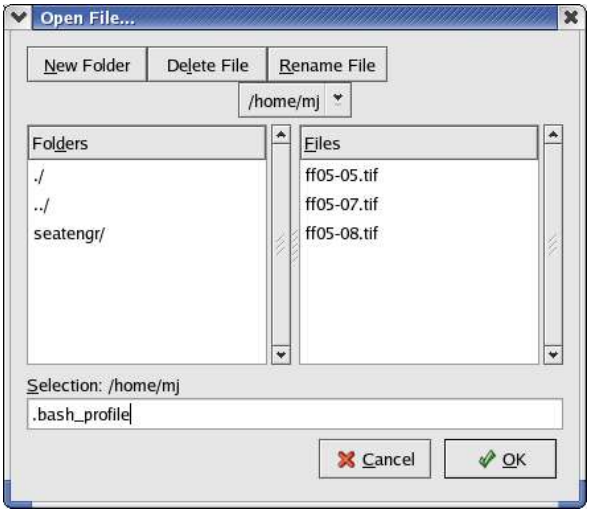


***Figure 9**. Opening the .bash_profile login configuration file.*

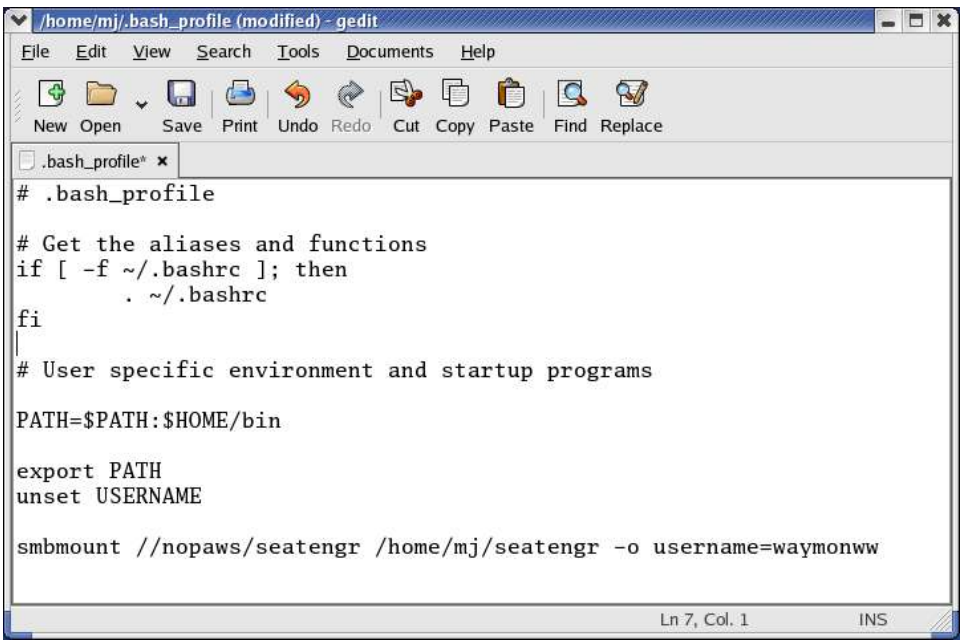


***Figure 10**. Editing the .bash_profile configuration file.*

Naturally, you can repeat this command, substituting the home directories of your choice for /home/padma. There are five things I need to explain about this command:

- The **echo** command copies the information within the quotation marks.
- The single quotation marks surround the command that I want to add to the .bash_profile file.
- The **$USER** variable defaults to the current user name.
- The backslash "escapes" the carriage return; in other words, Linux reads the previous command as if it were on a single line.
- The double forward directional arrow (>>) adds the contents within the quotation marks to the end of the target file.

All you need to change is the home directory of the target file; for example, to add this line to the .bash_profile file for user **nancy**, change the target file from /home/padma/.bash_profile to /home/nancy/.bash_profile.

If you have a lot of users, there are ways to build time-saving scripts that allow you to make the same change for all the users on your system. For more information, read the upcoming book from Hentzenwerke Publishing, *Linux Transfer for Power Users.*

Peer-to-peer Workgroups

Not all networks are configured in a Domain. In a peer-to-peer Workgroup, all computers can be configured as workstations and as servers. Just as there is no central server, there is no central database of user names and passwords, and therefore no need for the **winbind** daemon.

All you need to participate in a Workgroup are a list of the other computers in that Workgroup. If you don't want to share directories or printers, you don't need to set up the Samba server on your Linux workstation. Just use the **smbclient** and **smbmount** commands described earlier.

While share mode does not require a user name, it does require a password. If you're connecting to a shared directory from either a Windows NT/2000/XP or a Linux workstation, you'll also need a valid user name on the server. However, if you're connecting to a shared directory from a Windows 9x/ME computer, you just need the read-only or full-access password for the share.



As of this writing, the GUI Nautilus tool only supports browsing from a Linux member server or PDC on a Domain. It does not work when started from a workstation in a peer-to-peer Workgroup.

For example, assume a user named **polk** has logged in to a Linux workstation. That user can find a list of shared directories on the Windows 9x/ME Workgroup computer named **reuben** by using the **smbclient** command. He does not need a specific password when viewing shared directories and computers on a peer-to-peer Workgroup, as shown in **Figure 11**.

```
[mj@Delilah mj]$ smbclient -L reuben
added interface ip=172.22.30.103 bcast=172.22.30.255 nmask=255.255.255.0
Got a positive name query response from 172.22.30.40 ( 172.22.30.40 )
Password:

        Sharename      Type      Comment
        ---------      ----      -------
        MY DOCUMENTS   Disk      Documents from W98 Workstation
        IPC$           IPC       Remote Inter Process Communication

        Server               Comment
        ---------            -------
        ALLACCESS            Windows XP Pro Workstation
        DELILAH              Linux / Samba Workstation
        NOPAWS               Samba Server
        REUBEN               win98se

        Workgroup            Master
        ---------            -------
        DARKSTAR             ALLACCESS
[mj@Delilah mj]$
```

***Figure 11**. Viewing computers on a Workgroup.*

As you can see, I've set up a share for the My Documents directory on the **reuben** computer. This share can be mounted. For example, **polk** can use the following command to mount that directory on the /home/polk/shared subdirectory:

```
# smbmount '//reuben/My Documents' /home/polk/shared
Password:
```

Because this is a share from a Windows 9x/ME computer, **polk** can enter the password associated with read-only or full access on that shared directory. If the **reuben** computer recognizes the password, **polk** can reverse the process by using the following command:

```
smbumount /home/polk/shared
```



*If you're logged in to your Linux workstation as a regular user, and the **smbmount** or **smbumount** commands do not work, you probably did not assign appropriate permissions to these commands as described earlier. Log in as the root user and use the **chmod u+s** command on the **smbmnt** and **smbumount** scripts as described earlier in this chapter.*

If you're setting up your Linux workstation as a server on a peer-to-peer Workgroup, you'll also need to install the **samba** RPM package and configure the smb.conf configuration file. Remember from Chapter 3, you'll need to set at least the following variables:

```
workgroup = workgroupname
security = share
```

But even with share-level security on a Red Hat Linux server, you still need a user name and password. (The same is true for a Windows NT/2000/XP computer on a peer-to-peer

network.) You can use the commands described in the previous section to mount the shared directories of your choice. Because user names and passwords are required to connect to a Linux computer, the following option also works on a peer-to-peer Workgroup:

```
security = user
```

Setting up accounts

If you have Linux workstations on your Domain, the way that Samba works with user accounts depends on the operating system associated with the workstation, PDC, and Domain member server. This chapter assumes that you're using a Linux workstation. That leaves four possible scenarios:

- A Windows PDC and a Windows Domain member server
- A Linux PDC and a Windows Domain member server
- A Windows PDC and a Linux Domain member server
- A Linux PDC and a Linux Domain member server

I'll examine each of these configurations. The way you connect to a Windows Domain member server is the same, whether the Domain is governed by a Linux or a Windows PDC. I've therefore consolidated the first two cases into a single section. As you read the remainder of this chapter, remember that Linux workstations rely on Samba; they can't log directly in to a Linux or Windows PDC database.



It's possible to set up a single database for Linux user names and passwords with the Network Information Service (NIS) or the Lightweight Directory Access Protocol (LDAP). Coverage of these options requires advanced Linux skills and is beyond the scope of this book. For more information, see the respective HOWTO documents from the Linux Documentation Project at ***www.tldp.org/HOWTO/HOWTO-INDEX/howtos.html****.*

A PDC and a Windows Domain member server

You're just starting out with Linux, and aren't ready to set it up as a server on your network. In a Domain with a Linux or a Microsoft Windows PDC and a Microsoft Windows member server, the configuration is straightforward.

Amazingly enough, connections work in just the same way once you've replaced the Windows PDC with a Linux PDC on your network. The only visible difference is the lack of preconfigured custom users and groups, such as Replicators and Backup operators.

Assume user **donna** logs in to the Linux workstation. She can connect to shared directories on the PDC or Windows Domain member server using accounts on the PDC. She wants to access the "ststephdocs" shared directory from a Windows member server named **ststephen**. She doesn't have an account on the PDC; she wants to use **michael**'s account. All **donna** needs to do is run the following command:

```
# smbmount //ststephen/ststephdocs /home/donna/shared -o username=michael
Password:
```

This mounts the shared "ststephdocs" directory from the Windows Domain member server on the /home/donna/shared directory. **donna** has used **michael**'s user name to connect to that share. Once she's done, **donna** can unmount the share by using the following command:

```
# smbumount /home/donna/shared
```

As an administrator, assume you want to set up **donna** to mount this directory the next time she logs in. The file you change depends on whether **donna** starts Linux in text or graphical modes. If **donna** logs in to a text console, you can add the appropriate **smbmount** command to the /home/donna/.bash_profile file. If **donna** logs in to the GUI, you can add the appropriate **smbmount** command to the Sessions dialog shown back in Figure 7.

Windows PDC and Linux Domain member server

Assume that you're gaining more confidence with Linux, and are ready to set it up as member servers on your Domain. In a Domain with a Microsoft Windows PDC and a Linux Domain member server, the configuration is still fairly straightforward.

From the Linux workstation, you can connect to shared directories on the Windows PDC or Linux Domain member server using accounts on the PDC. This assumes that you've activated the Samba and Winbind dameons on the Linux Domain member server.

Assume user **nancy** logs in to the local Linux workstation. She wants to connect to the [shared] directory from the Linux Domain member server named **nopaws**. She doesn't have an account on the PDC; she wants to use **elizabeth**'s account. All **nancy** needs to do from the command line is to run the following command:

```
# smbmount //nopaws/shared /home/nancy/shared -o username=elizabeth
Password:
```

This mounts the [shared] directory from the Linux Domain member server on the /home/nancy/shared directory. The user **nancy** has used the account of user **elizabeth** to connect to that share. Once **nancy** is done working with that directory, she can unmount the share by using the following command:

```
# smbumount /home/nancy/shared
```

As an administrator, assume you want to set up user **donna** to mount this directory the next time she logs in. The file you change depends on whether **donna** starts Linux in text or graphical modes. If **donna** logs in to a text console, you can add the appropriate **smbmount** command to the /home/donna/.bash_profile file. If **donna** logs in to the GUI, you can add the appropriate **smbmount** command to the Sessions dialog shown back in Figure 7.

As an administrator, assume you want to set up user **nancy** to mount this directory the next time she logs in. The file you change depends on whether **nancy** starts Linux in text or graphical modes. If **nancy** logs in to a text console, you can add the appropriate **smbmount** command to the /home/nancy/bash_profile file. If **nancy** logs in to the GUI, you can add the appropriate **smbmount** command to the Sessions dialog back in Figure 7.

You can also add **elizabeth**'s password on the PDC so **nancy** does not have to enter passwords twice during the login process.



*If the connection does not work with PDC accounts, check the status of the Winbind daemon by using the **/sbin/service winbind status** command. If it's working, run the **wbinfo** commands described in Chapter 4, "Setting Up Your File Server's Users." These commands ensure that the Linux Domain member server is communicating with the PDC.*

Linux PDC and Linux Domain member server

Now assume that you're going further with your network, and you have Linux workstations, Linux Domain member servers, and a Linux PDC. In this particular configuration, Samba doesn't yet perform as well as Microsoft Windows Server on a Domain.

In this configuration, you can use Linux Domain accounts to browse the computers, shared directories, and printers for each member of that Domain. You can set up Microsoft Domain-style computer accounts on a Linux PDC. Unfortunately, you can't use the authentication database on a Linux PDC to mount shared directories from a Linux Domain member server. This capability probably won't be available until Samba 3.0 is released.



You could also set up a single database of user names and passwords for your Linux computers using the NIS or LDAP systems described earlier.

From the Linux workstation, you can connect to shared directories on the Linux PDC using the Domain accounts on the PDC. You can connect to shared directories on Linux Domain member servers using accounts on that server.

For example, if you've configured a user **waymon** on the Linux PDC, you can use **waymon**'s user name and password to browse the shares on Linux member servers. However, you can't mount a shared directory from the Linux member server unless you've added **waymon**'s user name and password to that member server.

Alternatively, assume that you're user **mj**. Log in to the local Linux workstation. There is no user **mj** on the Linux Domain member server. You're using the account of user **randy** on that member server. You want to access the shared directory named [shared] from the Linux member server named **nopaws**. From the command line, just run the following command:

```
# smbmount //nopaws/shared /home/mj/shared -o username=randy
Password:
```

This mounts the [shared] directory from the Linux Domain member server on the /home/mj/shared directory. The user **mj** has used **randy**'s user name to connect to that share. Once he's done, he can unmount the share with the following command:

```
# smbumount /home/mj/shared
```

If you want to set up user **mj** to mount this directory the next time he logs in, you can add the appropriate **smbmount** command to the /home/mj/bash_profile file for text logins. For

graphical logins, you can add the appropriate **smbmount** command to the Sessions dialog back in Figure 7.

Conclusion

You can set up a Linux workstation in a Microsoft Windows-style Domain or Workgroup. While you can use the GUI Nautilus tool or the text **smbclient** command to browse shared directories and printers, as an administrator you need to learn how to use text commands so you can set up logon scripts for individual users.

When a Linux user logs in, Linux runs a different logon script depending on whether the user logs in from the text or graphical console. You can set up text logon scripts by editing the hidden .bash_profile file in each user's home directory. You can set up GUI logon scripts with the Sessions manager in GNOME. Both require you to configure the **smbmount** and **smbumount** commands with appropriate permissions for all users.

You can also set up a Linux workstation in a peer-to-peer network. If you're connecting to a share from a Windows 9x/ME computer, all you need is the read-only or full-access password for that share. If you're connecting to a share from a Windows NT/2000/XP or another Linux computer, you need an authorized user name and password from that target computer.

In the next chapter, we'll take a look at configuring a Microsoft Windows workstation in a network with Linux servers.

> Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 6
Connecting Windows Workstations

Because this is a book about using Linux on a Microsoft Windows-based network, this chapter shows you how to connect various Microsoft Windows workstations to a Linux-based PDC. You've already configured a Linux-based PDC and member server in Chapters 3 and 4. In this chapter I'll show you how to connect Windows 9x/NT-style workstations to those computers in a Microsoft-style Workgroup or Domain.

If you're setting up workstations on a Domain, the first step is to recheck the computer serving as the Linux PDC. On that computer, you want to make sure that computer accounts, logon scripts, and profiles are ready to connect. You'll also want to record some basic network settings from the PDC to help you configure your Windows workstations.



This book is directed toward administrators of Microsoft Windows networks. If this describes you, just about everything in this chapter should seem familiar. A number of the instructions in this chapter will in fact seem elementary. However, Linux administrators also will read this book, and will therefore need extra help with handling Microsoft Windows workstations.

In this chapter, I'll show you how to set the network properties for each Microsoft Windows workstation. When you reboot, you can then connect and log in to the Domain through the Linux PDC. You can also set up roaming profiles if desired. I'm assuming that each workstation has a standard network card that has already been detected by your Microsoft Windows workstation.

If you've administered a Microsoft network before, many of the techniques in this chapter should be familiar to you. While most of this book is geared toward the experienced Microsoft administrator, this chapter is focused more toward the Linux administrator who is less familiar with Microsoft workstations.

Whatever operating system you use, you can observe and troubleshoot network communication on this Domain though Linux log files in the /var/log/samba directory.

Preparing accounts

This section is based on the work you did in Chapters 3 and 4 to configure a PDC on a Linux computer. Windows NT/2000/XP workstations (like Linux workstations) can't connect to a PDC unless they have a computer account on the PDC. Every user who is connecting to a Domain also needs a user account on that PDC.

As described in Chapter 4, computer accounts are stored in the /etc/passwd file, and are made available to Microsoft Windows networks through /etc/samba/smbpasswd. Once your Windows computer has connected to the Domain, you'll see accounts for that computer in these two files.

If you haven't yet connected your Windows workstation computer to the Domain, the **add user script** command in the smb.conf configuration file should help. By design, it adds the computer account to these files when you connect your workstation to the Domain. Alternatively, you can add the computer account manually. For more information, see Chapter 4, "Setting Up Your File Server's Users."

Logon scripts

Logon scripts are commonly used to automatically connect users to shared directories and printers. They should be located in the directory defined in the smb.conf [netlogon] share. They'll work for all Windows workstations as long as they're saved as MS-DOS text files with a BAT extension. You can do this by saving files in text format with Microsoft Windows WordPad, not Notepad.



Microsoft Windows WordPad adds a return character to files saved in text format that Notepad and Linux text editors, such as gedit, do not.

Thus, to configure Windows logon scripts, you'll need both a PDC and a Microsoft Windows computer on a network. Installing a PDC on a Linux computer requires Samba. In this case, I'll be setting up a script for a user named **pilot**. Once you have these computers available, take the following steps:

1. On the Linux computer, log in as the root user. Open the Samba configuration file, /etc/samba/smb.conf. Make sure to activate the [netlogon] share. I'm assuming that you're setting up the share as described in Chapter 4:

```
[netlogon]
  comment = Network Logon Service
  path = /home/netlogon
  guest ok = no
  writable = yes
```

2. In the Samba configuration file, activate the appropriate **login script** command. I'm assuming that you're activating scripts by user name, which corresponds to the following command:

```
login script = %U.bat
```

3. Reload the Samba configuration with the **/sbin/service smb reload** command. This makes Samba read the new smb.conf configuration file.
4. On a Microsoft Windows computer, log in as the administrative or root user on the Domain.
5. Open Microsoft Windows WordPad by clicking Start | Run and then typing **wordpad** in the text box that appears. Then press Enter.

6. In Microsoft Windows WordPad, enter the commands that you want for the user named **pilot**. For example, I add the following command to mount the [tmp] share from the **nopaws** computer on the L: drive:

```
net use L: \\nopaws\tmp
```

7. Once you've created your desired netlogon file, save the file in text format. For user **pilot**, you would save it as pilot.bat.
8. If you've logged in as the administrative or root user on the Domain, you should be able to save pilot.bat directly to the [netlogon] share on the PDC.
9. Return to the PDC. Restart Samba with the following command:

```
/sbin/service smb restart
```

While all users will have to log in to the Domain again, that's required before users can access a netlogon share.

Profiles

As described in Chapter 3, you can configure roaming profiles for Microsoft Windows workstations. Profiles for Windows 9x/ME computers are different from Windows NT/2000/XP computers and are stored in different locations. While Windows 9x/ME profiles are stored in users' home directories, Windows NT/2000/XP profiles are stored as defined by the [Profiles] share and **logon path** variable defined in your smb.conf file.

If you're converting from a Microsoft-based PDC to a Linux-based PDC as defined in Chapter 4, you can copy roaming profile files to the corresponding home directories. You can copy Windows 9x/ME profiles directly to users' home directories. For user **mj**, that's the /home/mj directory on the Linux PDC.



One of the drawbacks of roaming profiles for Windows 9x/ME computers on a Linux PDC is that users can accidentally delete their own profiles on their home directories.

For Windows NT/2000/XP profiles, there are two variables. In Chapter 4, we defined the **path** variable in the [Profiles] share as /home/profiles. We defined the **logon path** variable as **\\%L\Profiles\%U**, which means the profile is stored in the /home/profiles/mj directory.

Configuring the Microsoft workstation

There are two basic types of Microsoft workstations. The first type is based on the 16-bit Microsoft operating systems: Windows 95, 98, and ME. The connections you make from these operating systems don't require a computer account on the PDC, because Windows 9x/ME computers are actually not full members of a Domain. This does not change whether the PDC is on a Windows or a Linux computer.

The other type of Microsoft workstation is based on the 32-bit Microsoft operating systems: Windows NT, 2000, and XP. These workstations are more flexible in terms of user names and passwords.



Other Microsoft operating systems are available. Because Linux with Samba is intended as a substitute for the Microsoft Windows server operating systems, I don't configure the Microsoft Server operating systems in this book. Older Microsoft workstations, such as those based on MS-DOS and Windows for Workgroups, are rarely in use and are therefore not covered in this book.

In the following sections, I'll examine how to connect each major Microsoft workstation to a Domain. While the techniques are basically the same for all three 16-bit Microsoft operating systems, I highlight some variations in each system. While I focus on connecting to a Domain, I address connections to a peer-to-peer Workgroup later in this chapter.

Connecting a Windows 95/98/ME workstation to a Domain

The methods you use to connect Windows 95/98/ME computers to a Domain are basically the same for all three operating systems. In the following sections, I'll illustrate how you can connect to a network, set up roaming profiles, connect to shared directories, and share with other computers in the network.

Windows 95 and encryption

If you're concerned about network security, you should use encryption at least on the most critical items such as passwords. With the following command, the standard Samba configuration assumes that you're using an operating system that encrypts passwords:

```
encrypt passwords = yes
```

Passwords sent from the latest version of the Windows 95 operating system, known as OSR2, are encrypted. If you have a Windows 95 workstation, it's easy to find its version. On the Windows 95 desktop, right-click the My Computer icon. This opens the System Properties dialog. On the General tab shown in **Figure 1**, you should see 4.00.950 B, which corresponds to OSR2.



***Figure 1**. Checking the version of Windows 95.*

Disabling encryption

If you want to use older Microsoft Windows 95 operating systems on your network, the only option is to disable encryption on the PDC and all other workstations on the network. Samba includes Microsoft Windows registry files that you can use on various Microsoft workstations. On Red Hat Linux 9, the files described in **Table 1** are located in the /usr/share/doc/samba-2.2.7a/docs/Registry directory.



If you've upgraded to a later version of Samba 2.2, the directory name will change accordingly. If you still can't find these files, the following command should help:

```
rpm -ql samba | grep .reg
```

***Table 1**. Registry files for disabling encryption.*

File	Operating System
Win95_PlainPassword.reg	Windows 95 OSR2 (4.00.950 B)
Win98_PlainPassword.reg	Windows 98 (all versions)
WinME_PlainPassword.reg	Windows ME
NT_PlainPassword.reg	Windows NT Workstation
Win2000_PlainPassword.reg	Windows 2000 Professional/Windows XP Professional
WinXP_SignOrSeal.reg	Windows XP Professional for joining a Domain with a Linux PDC

To apply the registry file to a particular operating system, copy it to a temporary directory on the target workstation. When you double-click it through a medium such as Windows Explorer, the contents of the REG file are automatically merged with the Windows operating system registry.



Be careful when applying registry files to Microsoft Windows. Any REG file that you run takes effect immediately. I recommend that you back up the Windows registry file before applying the Samba revision.

Remember, if you want to use clear text passwords on a network with Windows 95 OSR2 computers (or later), you'll need to disable encrypted passwords on each workstation with the files as noted in Table 1.

There are two basic steps associated with setting up a Windows 9x/ME computer on a network. First you need to configure networking on your computer. Then you can configure a network connection to a Workgroup or Domain.

Configuring a connection to a network

To configure Windows 95/98/ME on a network, have your Windows 95/98/ME CD ready, and then take the following steps:

1. Right-click the Network Neighborhood icon on the desktop to open the Network dialog.

2. On the Configuration tab shown in **Figure 2**, make sure that you have installed at least the noted components, similar to what's shown in the associated text box.

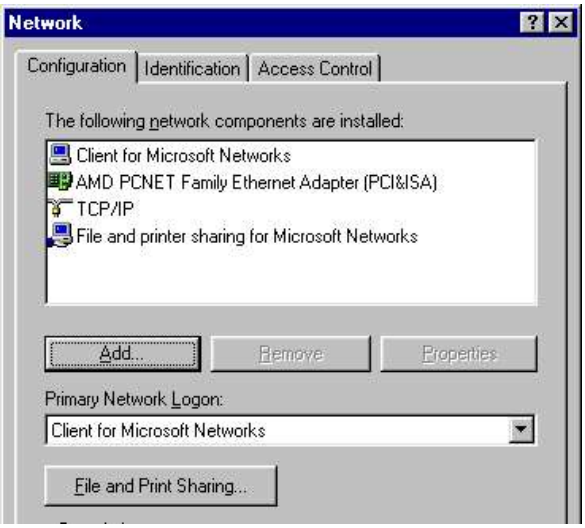


***Figure 2**. Windows Network Configuration.*

- **Client**: Client for Microsoft Networks supports login connections to a Domain.
- **Adapter**: A network adapter is required for a connection to a network; Windows normally detects network adapters upon installation. Hardware installation details on a Microsoft computer are beyond the scope of this book.
- **Protocol**: TCP/IP networking accommodates the default Linux network. If you have more than one TCP/IP protocol entry, make a note of the one associated with your network adapter.



This book assumes that you're using TCP/IP on your network, because that is the default for Linux and most Microsoft operating systems. If it isn't installed, click Add in the Network dialog shown in Figure 2.

- **Service**: "File and printer sharing for Microsoft Networks" allows you to share directories and printers from this computer.

3. Make sure the Primary Network Logon is set to Client for Microsoft Networks.
4. If any of these components are missing, click Add. In the Select Network Component Type dialog, select a missing component (Client, Adapter, Protocol, or Service), click Add, and then follow the prompts. Repeat as needed.

Configuring a connection to a Domain

Now I'll show you how to connect your configured Windows 9x/ME computer to a Domain. The following steps are essentially a continuation from the previous section, as I assume you still have the Windows Network configuration dialog open on your computer.

1. Click the File and Print Sharing button if you want to share directories or printers from this computer.
2. Highlight Client for Microsoft Networks and click Properties, which opens the window shown in **Figure 3**. If you're connecting this computer to a Domain, select the "Log on to Windows NT domain" check box, and then enter the name of the Domain governed by the Linux PDC. The network logon options allow you to reconnect automatically to any directories that you've shared before. Click OK when you've made your choices.

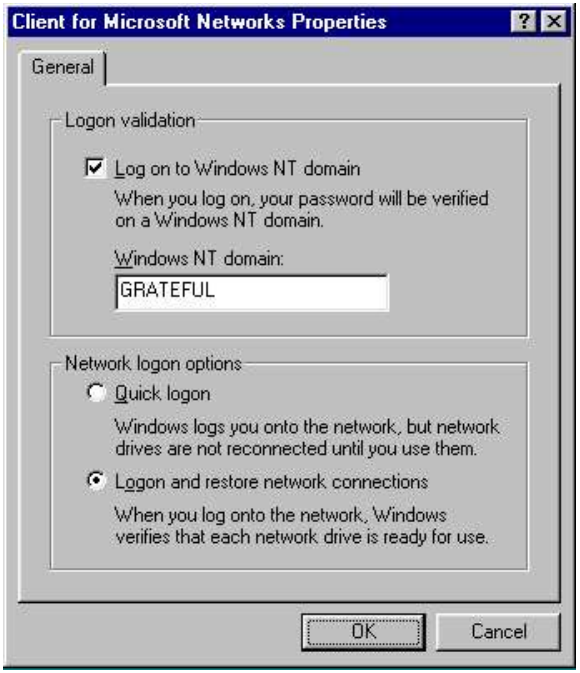


***Figure 3**. Setting up a connection to a Domain.*

3. You'll also need to set up your IP address information. Back on the Configuration tab, highlight TCP/IP. If you have more than one network adapter, be sure to select the TCP/IP setting associated with your network card, and then click Properties (see **Figure 4**).

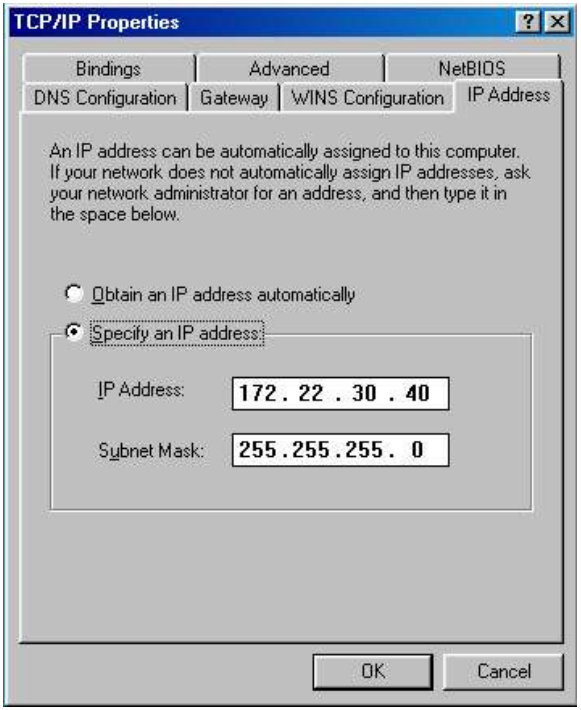


***Figure 4**. Setting a static IP address.*

4. If you aren't using a DHCP server, you'll need to add static IP address information. To do so, click the IP Address tab and then enter an IP address and subnet mask.
5. If you've configured a WINS server on the Linux PDC, click the WINS Configuration tab, and include the IP address of the PDC.



*You can enable WINS support on the Linux PDC by activating the **wins support** command in the smb.conf file as described in Chapter 3, "Setting Up Your Server File System."*

6. If you've set up a static IP address and need a connection to an external network such as the Internet, you'll need to add Gateway and DNS IP addresses on their respective tabs. Click OK when you're done.
7. Back in the Network dialog (Figure 2), click the Identification tab, where you can define three more things about this workstation:

 - **Computer name**: The NetBIOS name of the workstation; limited to 15 characters.
 - **Workgroup**: The name of the Workgroup or Domain that you're joining.

- **Computer Description**: A comment associated with this workstation.

8. Click the Access Control tab. If you're connecting to a regular peer-to-peer Workgroup, select "Share-level access control." If you're connecting to a Domain, select "User-level access control," and then enter the name of the Domain. Click OK when you're done.



If you're connecting to a Domain, you may get a message such as "Windows could not verify the specified security provider." If you do, click OK and wait a minute. It might take a short while before the Windows 9x/ME workstation finds the PDC.

9. If required, follow the prompts to insert your Windows 95/98/ME CD. Restart the computer when prompted.

10. After Windows 95/98/ME reboots, you'll get to log in to the Domain for the first time; **Figure 5** illustrates a login to the GRATEFUL Domain. (If you're connecting to a peer-to-peer Workgroup, the Domain option is not shown in Figure 5.)

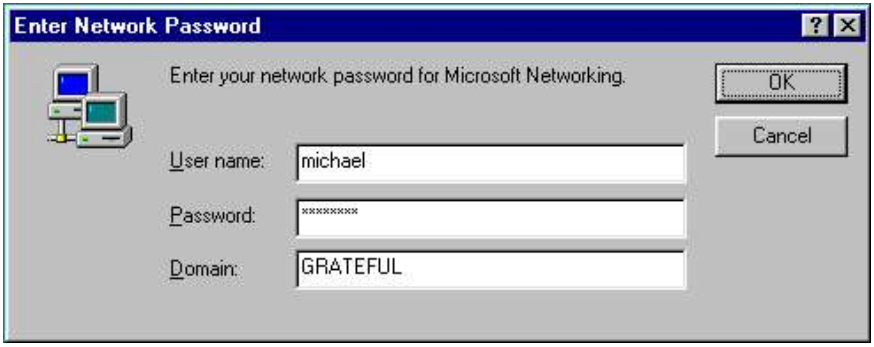


***Figure 5**. Logging in to the GRATEFUL Domain.*

Setting up roaming profiles

Roaming profiles allow users to get the same look and feel on their desktops regardless of which Windows 9x/ME workstation on a Domain they log in to. Roaming profiles are stored on the PDC, and when enabled, are sent over the network to the workstation. Unfortunately, Windows 9x/ME roaming profiles are not interchangeable with those available for Windows NT/2000/XP.

Not all administrators will want to create roaming profiles, because they can get quite large. For example, the roaming profile on my Windows XP Professional workstation is nearly 300MB. It takes quite a while to transmit that one profile from my PDC to my Windows XP Pro workstation on my Ethernet network. Network performance suffers during this process.

If your users have large roaming profiles and all log in at the start of a workday, that can easily tax the capacity of even faster networks. You may decide that it isn't efficient to set up roaming profiles for your users. In that case, you'll want to read the following with a view toward making sure that all user profiles are local.

To configure a roaming profile on a Windows 9x/ME computer, follow these steps:

1. Click Start | Settings | Control Panel.
2. Double-click Passwords in the Control Panel window.
3. Click the User Profiles tab. Make the selections shown in **Figure 6**.

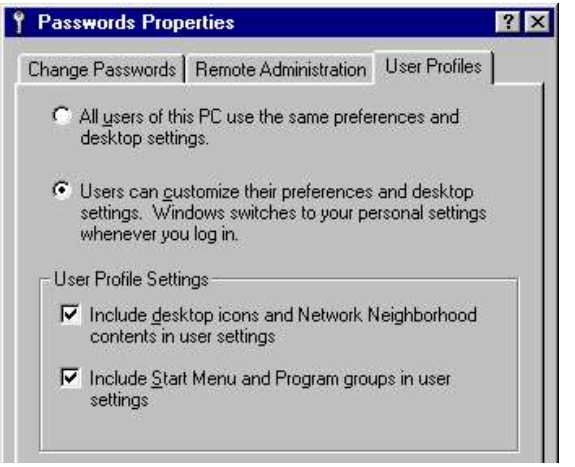


***Figure 6**. Setting up a roaming profile.*



If you want to disable roaming profiles, select the first option on the User Profiles tab in Figure 6: "All users of this PC use the same preferences and desktop settings."

4. Log off your system. Windows 9x/ME then transfers your user profile to the PDC. In the /etc/samba/smbusers file, Windows user **michael** is associated with Linux user **mj**; therefore, this profile is saved in the /home/mj directory.



If you're connecting to a peer-to-peer Workgroup, user profiles are stored locally. Because there is no central logon server, you can't set up a roaming profile.

Connecting to a share

There are three basic ways to connect to a shared directory or printer from a Microsoft Windows computer. You can connect graphically through Network Neighborhood, map a network drive to a known share, or connect from the MS-DOS prompt.

The most efficient way to view a Network Neighborhood is through Windows Explorer. To open it, click Start | Programs | Accessories | Windows Explorer. When you navigate to

Network Neighborhood, you can see the computers in your Domain. **Figure 7** shows the computers in my Grateful Domain.

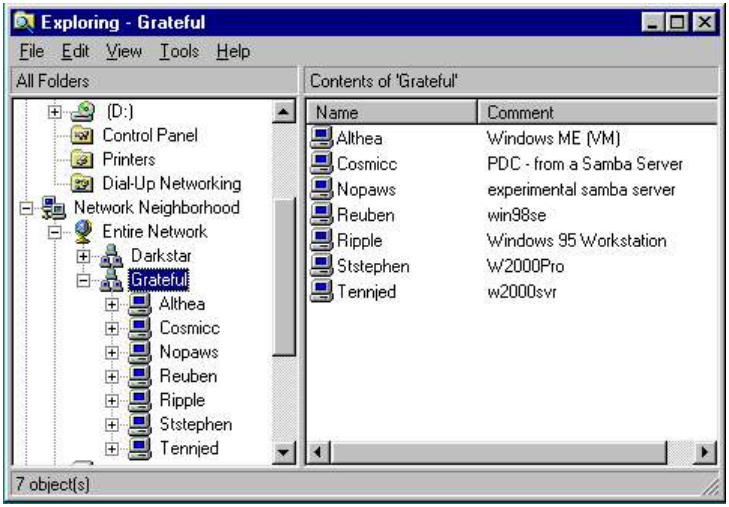


***Figure 7**. A Network Neighborhood view of a Domain.*

Once you've connected to a Domain via Network Neighborhood, highlight the computers of your choice. See what happens in the right-hand pane of Windows Explorer. You'll see the printers and directories that are shared and browseable from that computer.

As you can see in **Figure 8**, I've connected to my home directory on the PDC, where you can see the files associated with my Windows 9x/ME profile. As long as you have appropriate permissions through the smb.conf file and on the directory, you can use it just like any other directory on your workstation.

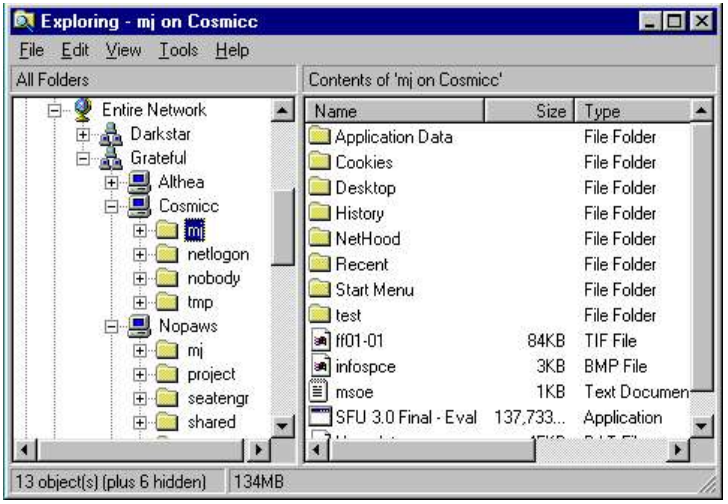


***Figure 8**. Using a shared directory from a Linux Domain member server.*

However, it can be cumbersome to drill down through Network Neighborhood to connect to a shared directory. A simpler option is to map that shared directory to a drive letter such as E:, F:, or G:. Once mapped, the drive letter will appear in Windows Explorer along with other local drives, which makes access to the shared directory much more convenient.

It's easy to specify a drive letter mount point. On the desktop, right-click My Computer and select Map Network Drive from the pop-up menu. In the Map Network Drive dialog, select from the available drive letters and set the path to the desired shared directory. For example, **Figure 9** illustrates how you would map drive E: to the same home directory (/home/mj) as shown in Figure 8.

Figure 10 illustrates the result in Windows Explorer.



***Figure 9**. Mapping to a mount point.*

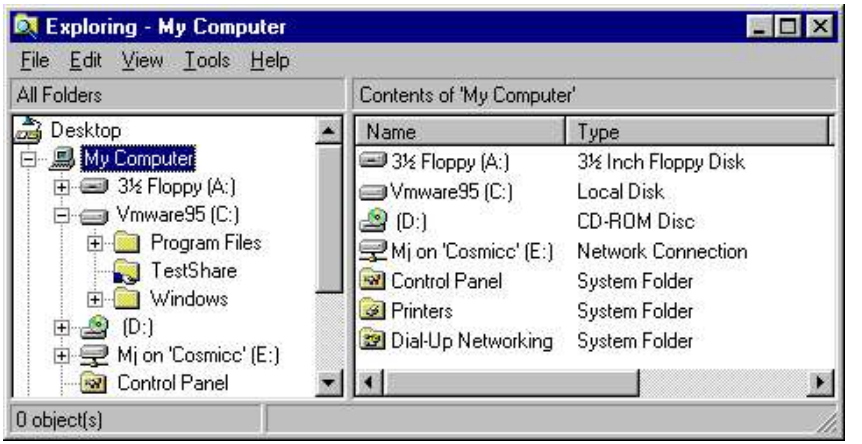


***Figure 10**. Regular and mapped drives.*

The shared directory is mounted to the noted drive letter, and is accessible as any other drive on your computer. Finally, you can mount a network drive in a similar fashion by using a text command at the MS-DOS prompt. Because the text commands apply to all Microsoft operating systems, I describe them in detail later in this chapter.



If you're prompted for a password, that's a problem on a Domain. If the user name and password you used to log in to Windows 9x/ME doesn't work for a share, the GUI prompt doesn't allow you to enter a different user name. You'll need to log out and log back in to the Windows 9x/ME computer with an appropriate user name.

Creating a Domain share

Now that you've logged in to a network, you can create a share from your workstation that's usable by other users on the Domain. To set up a share, point to a folder through Windows Explorer or My Computer. Right-click the folder that you want to share; if you've set up networking properly, you can select the Sharing command from the pop-up menu. This brings up a Properties dialog associated with the name of the directory that you want to share. As you can see in **Figure 11**, I've configured a directory on the local workstation named TestShare for access by several different users, as taken from the database of user names on the PDC.

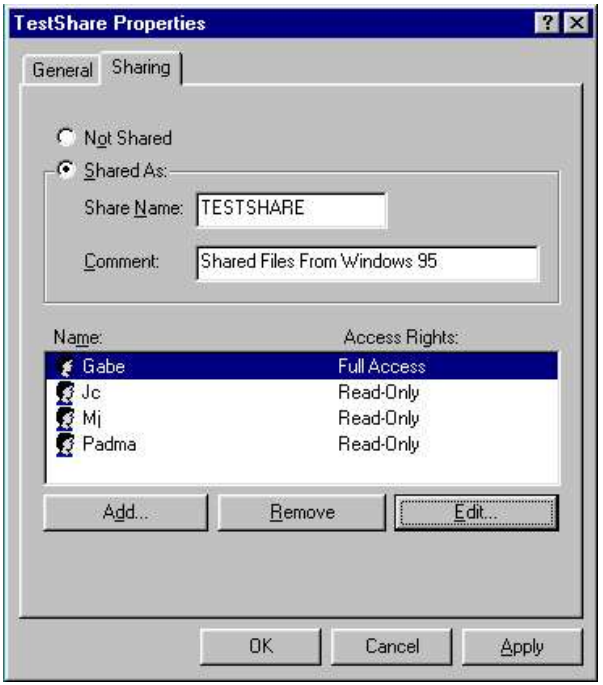


***Figure 11**. Sharing over a Domain from a Windows 95/98/ME directory.*

I described the nature of Linux permissions in Chapter 4, "Setting Up Your File Server's Users." As you can see, even the access rights that you can assign on a Windows 9x/ME computer are somewhat more fine-grained than what you can assign to a Linux file. However, the rights you set on a Windows 9x/ME workstation can be applied to users who log in from other Microsoft Windows computers on the network, even though the PDC is on a Linux computer.

Once you've granted access rights to certain users or groups, you can customize their rights. Highlight a user and click Edit. As you can see in **Figure 12**, you can customize the rights for individual users or groups. A detailed discussion of the rights that you can grant on a Windows computer is beyond the scope of this book.

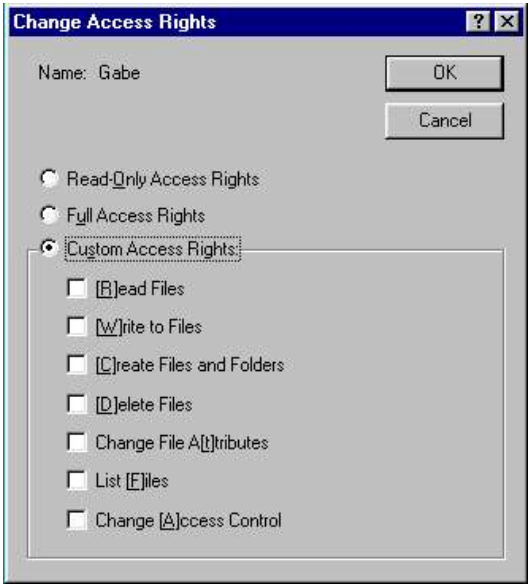


***Figure 12**. Customizing user access rights to a shared directory.*

Special Windows 98 issues

There are no basic problems with connecting Windows 98 computers to a peer-to-peer Workgroup or a Domain with a centralized database of user names and passwords. Windows 98 computers by default send passwords to the PDC in encrypted format. Naturally, Windows 98 includes drivers for more network adapters than are available on the Windows 95 CD.

Otherwise, there are few significant differences between Windows 98 and the other 16-bit Microsoft operating systems, at least with respect to network connections. Here are a couple of items that you might find useful:

- When you view Windows 98 network properties similar to Figure 2, Windows 98 always includes a default "Dial-Up Adapter" even if you don't have a telephone modem. Just be careful to configure the network card and not the modem.
- If you want to keep older (pre-OSR2) Windows 95 computers on your network, you'll need to set up clear text (non-encrypted) passwords on your network. While I don't recommend this procedure, it may be acceptable if your network is protected by a firewall. You can set up clear text passwords on a Windows 98 computer by applying the Win98_PlainPassword.reg file described in Table 1 to your Windows 98 registry.

Special Windows ME issues

There are no basic problems with connecting Windows ME computers to a Workgroup or Domain. Windows ME computers normally transmit passwords in encrypted format.

Functionally, Windows ME is very close to Windows 95/98. However, there are some significant differences in the look and feel of this operating system. For example:

- Windows ME includes My Network Places instead of Network Neighborhood. You can right-click the My Network Places icon to configure a connection to a Domain or a Workgroup. Once you see the Network Properties dialog, the actions you take are identical to Windows 95/98.
- If you double-click and open My Network Places, you can run the Add Network Places wizard to browse and connect to available shared directories.
- In My Network Places, the Home Networking wizard can help you configure your computer on a Domain or a peer-to-peer Workgroup. It can also help set up a connection through a network gateway to a remote network such as the Internet.
- If you're setting up clear text (non-encrypted) passwords on your network to accommodate older, pre-OSR2 Windows 95 computers, apply the WinME_PlainPassword.reg file described in Table 1 to your Windows ME registry.

Creating a Windows 95/98/ME Workgroup share

If you're setting up a peer-to-peer Workgroup, the steps you'll take to share a directory from a Microsoft Windows 95/98/ME workstation are slightly different. The main difference is that peer-to-peer Windows Workgroups use only passwords. User names are not required to connect to a shared directory from these 16-bit operating systems.

To include a Linux computer as a Samba server on a Windows Workgroup, you'll need to change at least the following variable in the /etc/samba/smb.conf configuration file:

```
security = share
```

Naturally, you'll need to change other variables, such as making sure that the **workgroup** variable is set to the name of the peer-to-peer Workgroup instead of a Domain. I describe the differences in more detail in Chapter 5, "Connecting Linux Workstations."

Creating a peer-to-peer Workgroup share on a Windows 9x/ME directory is a fairly straightforward process. Open Windows Explorer by clicking Start | Programs | Accessories | Windows Explorer, and then right-click the directory that you want to share. In the pop-up menu that appears, click Sharing. This should open a Properties window named for the directory that you're sharing, similar to **Figure 13**.



If you don't see Sharing in a pop-up menu when you right-click a folder, click Properties. If you can share a directory, click the Sharing tab. If you don't see a Sharing tab, you need to configure networking as described earlier in this chapter.

You can set up different passwords to share files from the folder. One password would support read-only sharing; a second password would support full-control shares, where connecting users have full control over the files and directories. If you set up both types of shares, you must use different passwords.

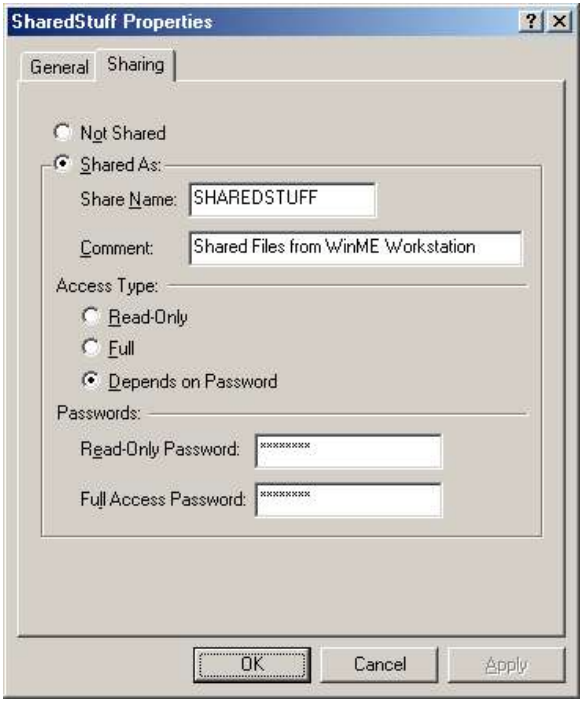


***Figure 13**. Setting up a peer-to-peer share in a Workgroup.*

Windows NT 4 Workstation

In this section, I'll show you how to connect a Windows NT 4 Workstation to a Domain governed by a Linux-based PDC. To this end, I'll illustrate how you can connect this 32-bit workstation to a network, set up roaming profiles, connect to shared directories on the server, and share its own resources with other computers in the network.

If you're setting up clear text (non-encrypted) passwords on your network to accommodate older (pre-OSR2) Windows 95 computers, apply the NT4_PlainPassword.reg file described in Table 1 to your Windows NT 4 Workstation registry.

There are two basic steps associated with setting up a Windows NT 4 Workstation computer on a network. First you need to configure networking on your computer. Then you can configure a network connection to a Workgroup or Domain.

Configuring a connection to a network

To configure a Microsoft Windows NT 4 Workstation on a network, have your Windows NT 4 Workstation installation CD ready, and then take the following steps:

1. Log in to NT 4 Workstation, using an account with administrative privileges.
2. Right-click the Network Neighborhood icon, and then click Properties. This opens the Network dialog shown in **Figure 14**.

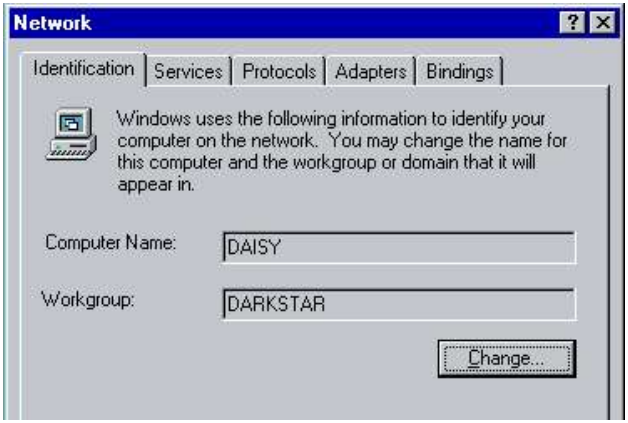


***Figure 14**. Configuring Windows NT 4 Workstation network services.*



If you don't see a Network Neighborhood icon in Microsoft Windows NT 4 Workstation, click Start | Settings | Control Panel. In the Control Panel window, click Network. (These steps also happen to work in Windows 95/98.)

3. Before you connect this computer to the Domain, you may want to check its network settings on the other tabs shown, which are described in **Table 2**.

***Table 2**. Network properties configuration tabs.*

Tab	Function
Identification	Lists the current computer name and Workgroup or Domain.
Services	Specifies services to be installed with the network protocol suite. You should have at least the Computer Browser, NetBIOS Interface, RPC Configuration, Server, and Workstation services installed; you can add them by using this tab, as long as you have the Windows NT 4 installation CD.
Protocols	Allows you to configure TCP/IP (or some other protocol stack) on this computer. More on this shortly.
Adapters	Lets you install and configure any network adapters on this computer.
Bindings	The NetBIOS, Server, and Workstation services are all normally bound as a WINS client to your network adapter.

4. Unless you have a DHCP server, you'll want to configure your TCP/IP settings manually on the Protocols tab. If you don't see those settings on that tab, you'll need to click Add and follow the prompts to add them. When you see TCP/IP Protocol, highlight it and then click Properties. As you can see from **Figure 15**, I've configured the manual IP address that I'm using on my Windows NT 4 Workstation computer.

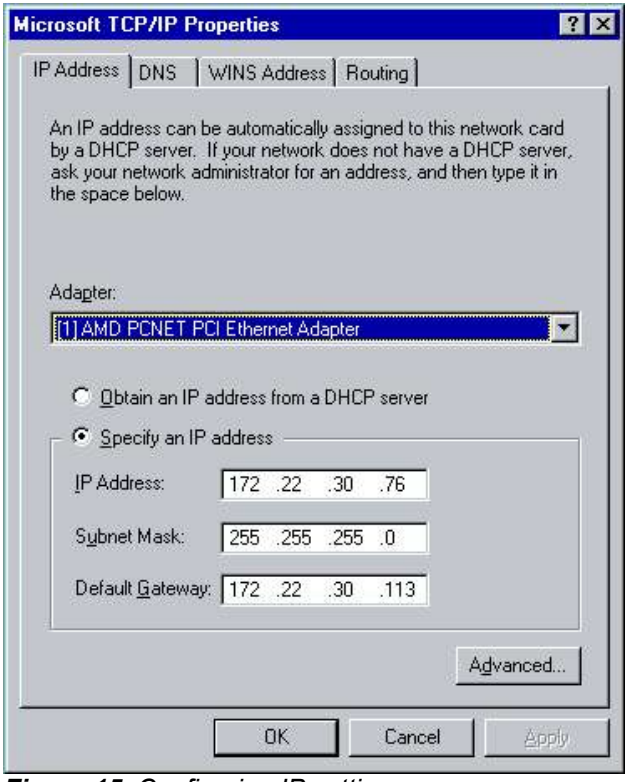


***Figure 15**. Configuring IP settings.*

You can configure other settings:

- Adapter: If you have more than one network adapter on this computer, you'll want different settings for each adapter. Different network adapters are frequently connected to different networks.
- If you have a DHCP server on your network, it's usually best to let it assign your IP address.
- If you click the Advanced button, you can configure different gateways as well as elementary firewalls.
- On the DNS tab, you can configure the DNS servers for your network.
- On the Routing tab, you can set up this computer as a router, a computer that allows two computers to communicate with each other.



If you have a DHCP server, it should at least assign an IP address, subnet mask, and a gateway address for each computer on your network. You can override your DHCP server by configuring your TCP/IP settings manually.

5. When you're finished configuring your IP address, click OK, and then return to the Identification tab of the Network dialog shown in Figure 14.

Configuring a connection to a Domain

Now that you've set up your NT 4 Workstation connection to a network, you can set it up for a connection to a Domain. To do so, let's go back to the Identification tab of the Network dialog.

1. Click Change to open the Identification Changes dialog shown in **Figure 16**.

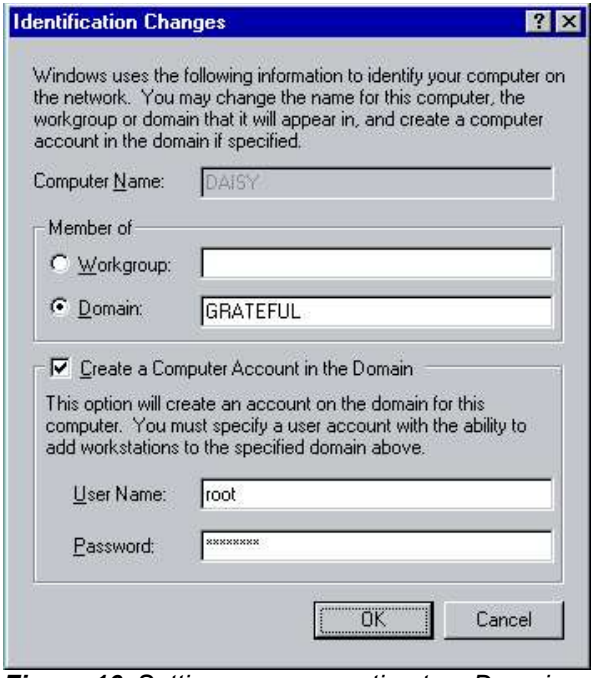


***Figure 16**. Setting up a connection to a Domain.*

2. To connect to a Domain, enter its name in the appropriate text box.
3. All Linux and Windows NT/2000/XP computers that connect to a Domain require a computer account on the PDC. You can create that computer account when you connect to the Domain for the first time. Note that you'll need the name and password of the administrative account on the PDC. (This assumes you've configured the **add user script** command in the PDC's smb.conf file as discussed in Chapter 4.)
4. If successful, you'll see a message welcoming you to the Domain that you selected. Click OK as needed to close the Network dialog.
5. Reboot the computer when prompted. Test the connection by logging in as a user on the Domain. Next, you'll see how you can implement roaming profiles on this Windows NT 4 Workstation computer.

6. Click Start | Shut Down. In the Shut Down Windows dialog, select "Close all programs and log on as a different user," and then click Yes.

Setting up a roaming profile

Roaming profiles allow users to get the same look and feel on their desktops, regardless of which Windows computer they use to log in to the Domain. Unfortunately, Windows 9x/ME roaming profiles are not interchangeable with those available for Windows NT/2000/XP.

As described earlier, roaming profiles can get quite large. For example, the roaming profile on my Windows XP Professional workstation is nearly 300MB. If your users have large roaming profiles, that can easily tax the capacity of many networks. In that case, you'll want to read the following with a view toward making sure that all user profiles are local.

If you want to configure a roaming profile on a Windows NT 4 Workstation computer, follow these steps:

1. Log in with an administrative account for the local computer (not the Domain).
2. Right-click the My Computer icon, and then click Properties to view the System Properties dialog.
3. Click the User Profiles tab. You can adjust the profiles of all users who've logged in to this particular workstation, as shown in **Figure 17**. As you can see, I've selected my Domain user **waymon**.

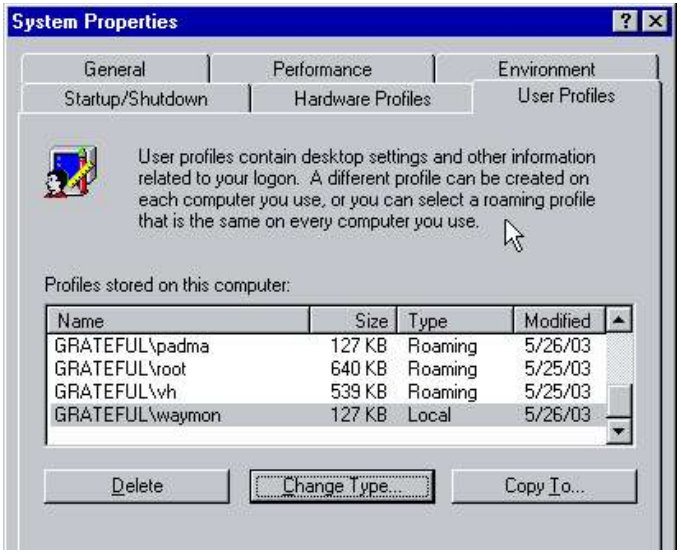


***Figure 17**. Local and roaming user profiles.*

4. Select a Domain user and click Change Type. As shown in **Figure 18**, you can store the profile locally. If you set a roaming profile, the profile is stored on the PDC, as defined by the [Profiles] share in the PDC's smb.conf file.

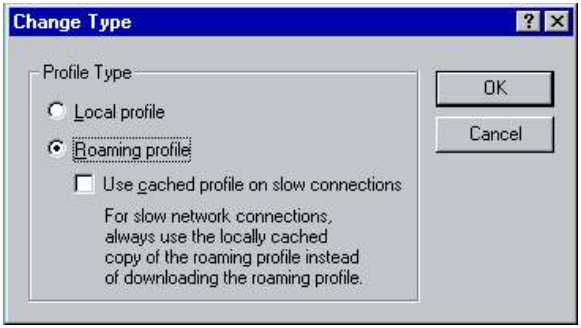


***Figure 18**. Switching between local and roaming profiles.*



Users who connect to a Domain over a slow connection such as a telephone modem should store profiles locally. If a roaming profile is desired, select the option "Use cached profile on slow connections."

5. Click OK to complete your changes. This saves the profile on the local computer. The next time **waymon** logs in to and out from this workstation, the profile is sent to and saved on the PDC computer.

Connecting to a Domain share

The easiest way to connect to a shared directory is through Windows Explorer. You can also mount the shared directory to a Microsoft Windows drive letter. Later in this chapter, I'll show you how to mount to a shared directory from the command-line interface.

To access the Windows Explorer file browser, click Start | Programs | Windows NT Explorer. (Depending on your profile, you may need to click Start | Programs | Accessories | Windows Explorer.)

You can navigate to the shared directories on the Grateful Domain through the Network Neighborhood, as shown in **Figure 19**. As you can see, I've connected to the share named "shared," which permits user access as defined in Chapter 4. The **nopaws** server is a Linux computer, configured with Samba as a Domain member server.



The first time you connect to a shared directory though a Windows workstation, you may have trouble connecting to a share on which you think you have permissions. While your workstation may have found the PDC, the PDC may not have found your computer yet on the browse list. Before doing anything else, try logging off and logging back in to that workstation.

It's easy to specify a drive letter mount point. Right-click My Computer and select Map Network Drive from the pop-up menu. In the Map Network Drive dialog, select from the available drive letters. Windows NT is a bit different from Windows 9x/ME in that you can browse the available shares, as shown in **Figure 20**.

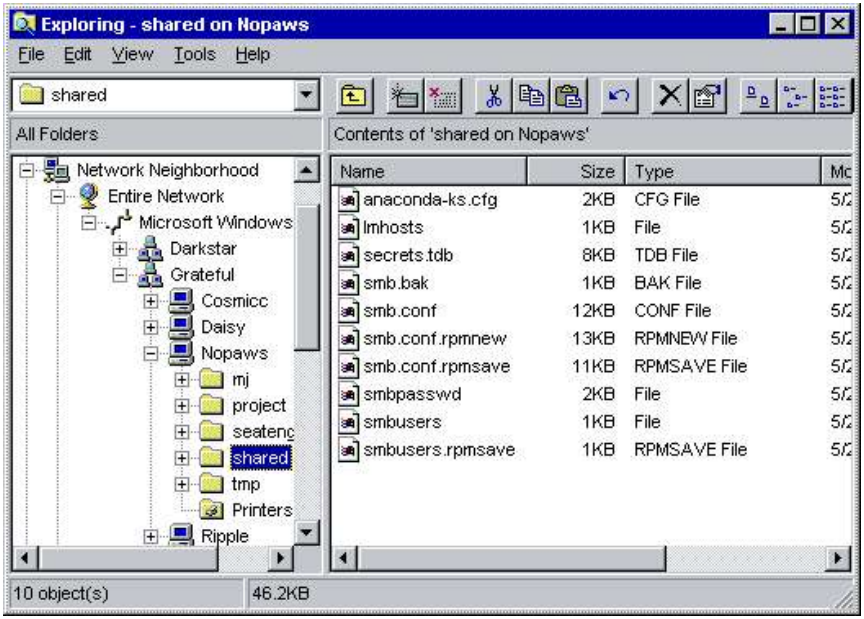


***Figure 19**. Navigating to a shared directory.*

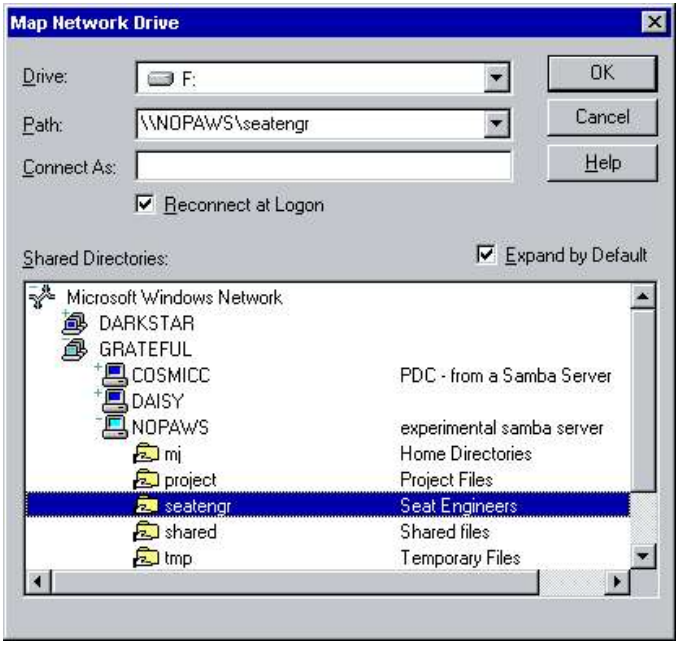


***Figure 20**. Mapping from Windows NT 4 Workstation to a mount point.*

When you highlight a specific share from a computer on the Domain, the path is automatically shown in the Path text box. Figure 20 illustrates a connection to the seatengr share on the **nopaws** Domain member server.

If the connection does not work, you can enter a different user name in the Connect As text box. If the user name corresponds to one on your Domain, you'll be prompted for a password.

The shared directory is mounted to the noted drive letter, and is accessible as any other drive on your computer. Whether you can read from or copy to the share depends on the permissions set on the Linux Domain member server in the governing smb.conf file.

Finally, you can mount a network drive in a similar fashion from the MS-DOS prompt. Because the text commands apply to all Microsoft operating systems, I describe them in detail later in this chapter.

Connecting to a Workgroup share

If you've configured a Windows NT 4 Workstation computer on a peer-to-peer Workgroup, the process for connecting to a share depends on whether it's from a Windows 9x/ME or a Windows NT/2000/XP computer. In either case, open Windows Explorer (or Windows NT Explorer) as described earlier. When you select a shared directory, Windows prompts for a user name and password.

For connections to a Windows 9x/ME share, you don't need a user name to connect to a shared directory. When you see the Enter Network Password dialog shown in **Figure 21**, you don't need to enter a user name. Just enter the read-only or full-access password associated with the shared directory and click OK.



***Figure 21**. Connecting to a shared directory on a peer-to-peer Workgroup.*



Don't worry about the error message shown in Figure 21. It just means that the password you used to log in to the workstation is different from the read-only or full-access password for the share. If the password does match, you won't see the Enter Network Password dialog, and will automatically get the permissions associated with the password.

On a Windows NT/2000/XP share, connections depend on the allowed users on the target computer. If share access is limited to a specific user name, you need to log in with the same user name on the local computer.

For example, I have a share named Downloads on my Windows XP Professional computer named **allaccess**. I limit access to the Domain user named **michael**, as configured on **allaccess**. If I log in to my Windows NT 4 Workstation computer as a Local or Domain user named **elizabeth**, I won't have access to the \\allaccess\downloads shared directory.

Creating a Domain share

Now that you've logged in to a network, you can create a share that's usable by other users on the Domain. To set up a share, point to a folder through Windows Explorer or My Computer. Right-click the folder that you want to share.

If you've set up networking properly, you'll be able to select the Sharing command in the pop-up menu. This brings up a Properties dialog associated with the name of the directory that you want to share. Make any changes and then click Permissions. As you can see in **Figure 22**, I've configured a directory named ShareTest for access from several different users, as taken from the database of user names on the PDC.

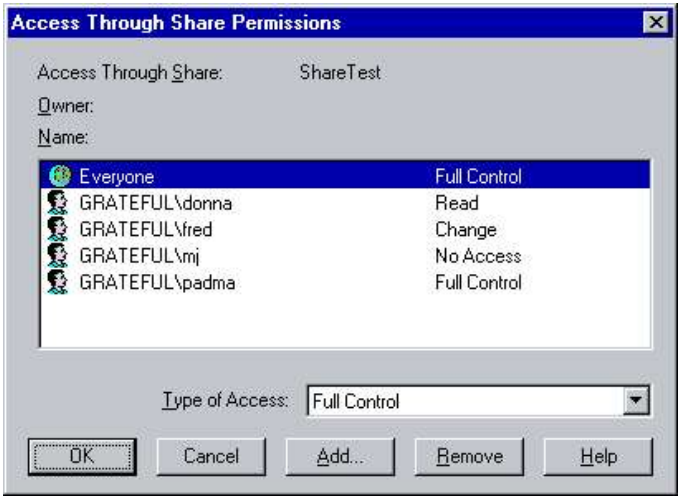


***Figure 22**. Setting up share access for specific users.*

If you want to add more users or groups to the share list, click Add. In the Add Users and Groups dialog, you can add more users or groups from your Domain.

On the Security tab, you can also set permissions, audit rights, and ownership for the subject directory. Details of this process are extensive and are beyond the scope of this book.

Creating a Workgroup share

You can set up shared directories from Windows NT 4 Workstation on a peer-to-peer Workgroup. To set up a share, log in to the workstation with an administrator account. The remaining steps are virtually identical to creating a share for a Domain.

The only difference is that the users that you can add to the permissions list are limited to those configured on the local NT 4 Workstation computer. By definition, there is no centralized list of users on a PDC. For example, assume I've given permissions to a shared directory named ShareTest to the user named **michael** on an NT 4 Workstation named **daisy**. Also assume I've deleted the "Everyone" entry from the Access Share Through Permissions dialog shown in Figure 22.

Other computers in the Workgroup can connect to this Workgroup share. From a Linux workstation, I'd mount this share on the /mnt/source directory with the following command:

```
# smbmount //daisy/sharetest /mnt/source -o username=michael
```



*Before a regular user can use the **smbmount** command, you need to set appropriate permissions as described in Chapter 5, "Connecting Linux Workstations."*

From a Windows workstation, I can connect if I can log in with a *local* user account named **michael**. I'd create that account if it didn't already exist.

Then after I log in, and I navigate to the share through Network Neighborhood, the local Windows workstation automatically passes **michael**'s user name and password to the share on the NT 4 Workstation named **daisy**.

Windows 2000 Professional

In this section, I'll show you how to connect a Windows 2000 Professional workstation to a Domain governed by a Linux-based PDC. To this end, I'll illustrate how you can connect this 32-bit workstation to a network, set up roving profiles, connect to shared directories, and share with other computers on the network.

If you need to accommodate older Windows 95 (pre-OSR2) computers on your network, you can set up clear text (non-encrypted) passwords by applying the Win2000_PlainPassword.reg file described in Table 1 on your Windows 2000 Professional workstation registry.

There are two basic steps associated with setting up a Windows 2000 Professional computer on a network. First, you need to configure networking on your computer. Then you can configure a network connection to a Workgroup or Domain.

Configuring a connection to a network

To configure a Microsoft Windows 2000 Professional workstation on a network, have your Windows 2000 Professional installation CD ready, and then take the following steps:

1. Log in to Windows 2000 Professional, using an account with administrative privileges on the *local* computer.
2. Right-click the My Network Places icon, and then click Properties. This opens the Network and Dial-up Connections window.
3. If you see a Local Area Connection icon, double-click it. This should open the Local Area Connection Status dialog. Click the Properties button to open the Local Area Connection Properties dialog shown in **Figure 23**.



If you don't see a Local Area Connection icon in the My Network Places window, you may need to reinstall the network card on your computer.

4. Select Internet Protocol (TCP/IP), and then click Properties. This opens the Internet Protocol (TCP/IP) Properties dialog.
5. If you don't have a DHCP server on your network, you'll need to enter your IP address and DNS server information in the appropriate text boxes.
6. Click the Advanced button to open the Advanced TCP/IP Settings dialog. Click the WINS tab, where you can enter information for the WINS server(s) on your network.

7. If needed, use the other tabs to add more information that you might need. The function of each tab is described in **Table 3**. When you're satisfied with the settings, click OK.

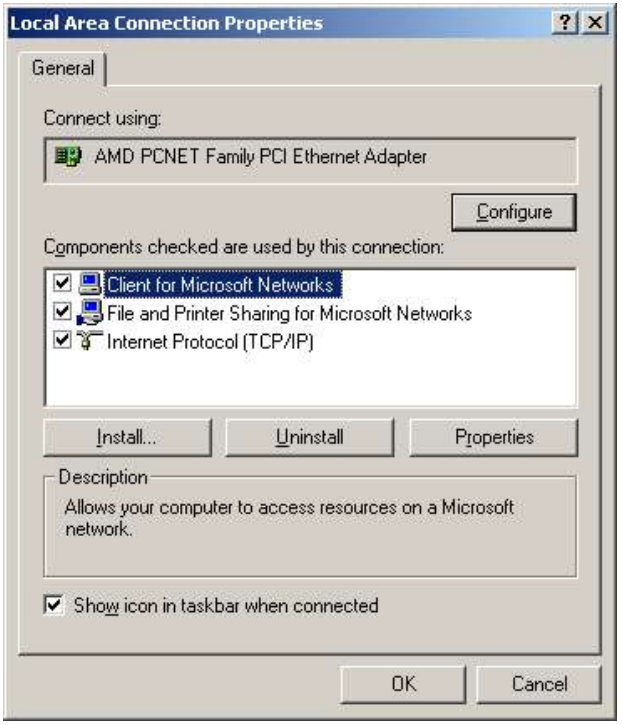


***Figure 23**. Configuring Windows 2000 Network services.*

***Table 3**. Tabs in the Advanced TCP/IP Settings dialog.*

Tab	Function
IP Settings	Lets you assign additional IP addresses and default gateways to this particular network card.
DNS	Allows you to specify any additional DNS servers that you might need, such as those specified by your network's ISP.
WINS	Supports configuration of a WINS server for this computer; to support connections to a Linux-based PDC, be sure to select the "Enable NetBIOS over TCP/IP" option.
Options	Permits configuration of a firewall on this computer.



Unlike Windows 9x/ME or NT, changes that you make to many network properties of a Windows 2000 Professional computer are incorporated immediately and generally do not require you to reboot your computer.

Configuring a connection to a Domain

Now that you've set up your Windows 2000 Professional computer connection to a network, you can set it up for a connection to a Domain, using the following steps:

1. Right-click the My Computer icon. In the pop-up menu that appears, click Properties to open the System Properties dialog. Click the Network Identification tab shown in **Figure 24**. As you can see, this computer is currently named **ststephen**, a member of the DARKSTAR Workgroup.



***Figure 24**. Configuring Windows NT 2000 network services.*



The options available to connect a Windows 2000 or a Windows XP Professional computer to a Domain are nearly identical. In this section, I'll show you the Network Identification Wizard. In the corresponding section on Windows XP Professional, I'll show you how to do the same thing with the Properties button.

2. Click the Network ID button to start the Network Identification Wizard.
3. When the Network Identification Wizard opens, click Next.
4. You're asked whether this computer is part of a business network or for home use. To accommodate the use of Linux computers on this network, select the business network option and then click Next.



If you want to set up this Windows 2000 computer on a peer-to-peer Workgroup or a Domain, select the business network option.

In the following step, I'm assuming that you want to make your computer a member of a Domain.

5. If you choose the other option (Properties), the Network Identification Wizard allows you to make this computer a member of a Workgroup. You'll just need to name the Workgroup and follow the prompts until you reboot this computer.
6. If you're making this Windows 2000 computer a member of a Domain, you'll need the information described in **Figure 25**. Click Next after you've collected this information.



***Figure 25**. A list of data you need to join a Domain.*

7. In the following screen, enter a user name and password on the PDC (not the workstation), and the name of your Domain. Click Next.
8. If you haven't connected this computer to the PDC before, you'll need to create a computer account on the Domain. Enter the name of your computer and the Domain. Click Next.
9. The Network Identification Wizard prompts you for an administrative account on the PDC. The standard administrative account on a Linux computer is root. You'll need to enter **root**, the root user password, and the name of the Domain. Assuming you set up the **Add User Script** command in the PDC's smb.conf file as described in Chapter 4, Linux should then create a computer account for you on the PDC.
10. Next, you can set up a Domain user for access to resources on this computer, as shown in **Figure 26**. After you've added a user from the Domain, click Next.
11. Now you can assign the user various levels of access. For example, I can set **donna** to be a standard power user, or a restricted user. Alternatively, if I select Other, I can select from one of the groups described in **Table 3**.

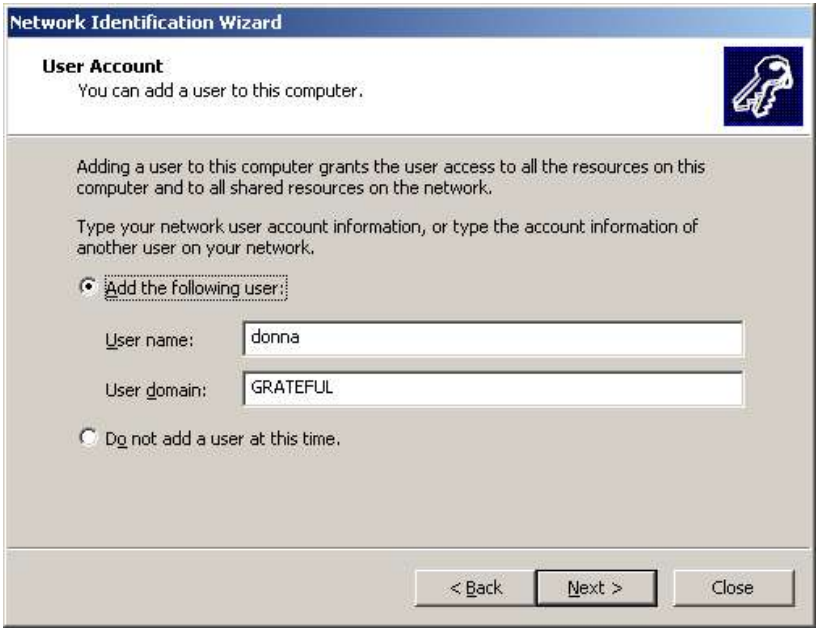


***Figure 26**. Adding a user.*

***Table 3**. Microsoft Windows user categories.*

Category	Function
Standard User	Allowed to log in, modify computer settings, and install applications on the local computer.
Restricted User	Allowed to log in, and read and write files.
Administrators	Given complete access to the Domain.
Backup Operators	Given full access solely for backing up or restoring files.
Guests	A restricted user, with additional limits.
Replicator	Domain users who are allowed to copy directories from computer to computer.

12. Click Next to complete the Network Identification Wizard. When you click Next once again, and then click OK to close the System Properties dialog, you'll be prompted to reboot this workstation. When you reboot, this computer will be connected to the Domain.
13. Test the connection by logging in as a user on the Domain. Next, you'll see how you can implement roaming profiles on a Windows 2000 Professional computer.
14. Click Start | Shut Down. In the Shut Down Windows dialog, select "Log off *username*," and then click OK.

Setting up a roaming profile

Roaming profiles allow users to get the same look and feel on their desktops when they log in from any Windows NT/2000/XP workstation on a Domain. Unfortunately, Windows 9x/ME roaming profiles are not interchangeable with those available for Windows NT/2000/XP.

Not all administrators will want to create roaming profiles, because they can get quite large. For example, the roaming profile on my Windows XP Professional workstation is nearly 300MB. If your users have large roaming profiles, that can easily tax the capacity of many networks. In that case, you'll want to read the following with a view toward making sure all user profiles are local.

To configure a roaming profile on a Windows 2000 Professional computer, follow these steps:

1. Log in with the user account of your choice from the Domain.
2. Right-click My Computer, and then click Properties. This opens the System Properties dialog.
3. Click the User Profiles tab. Roaming profiles are enabled by default, as shown in **Figure 27**. As you can see, I've logged in as the Domain user **mary**.

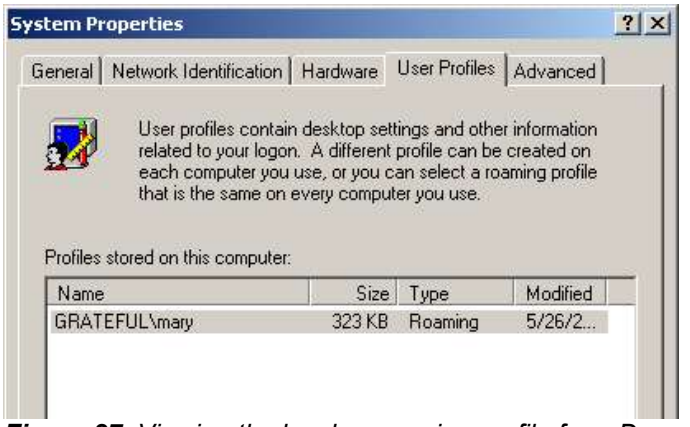


***Figure 27**. Viewing the local or roaming profile for a Domain user.*

4. Select the listed user and click Change Type to open the Change Profile Type dialog, which allows you to switch the configuration for this Domain user between a local and a roaming profile.

Connecting to a Domain share

The easiest way to connect to a shared directory in Windows 2000 is through My Network Places in Windows Explorer. You can also mount the shared directory to a Microsoft Windows drive letter. Later in this chapter, I'll show you how to mount to a shared directory from the command-line interface.

To access the Windows Explorer file browser, click Start | Programs | Accessories | Windows Explorer. You can navigate to the shared directories on the Grateful Domain through

My Network Places, as shown in **Figure 28**. In this case, you can see the different directories shared from the **nopaws** computer, configured with Samba as a Domain member server.

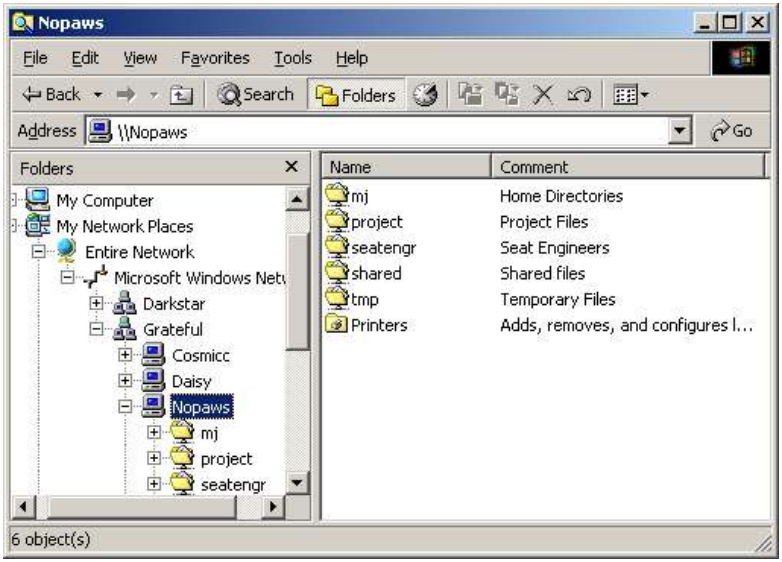


***Figure 28**. Navigating to a shared directory.*



The first time you connect to a shared directory though a Windows workstation, you may have trouble connecting to a share where you think you have permissions. You may be prompted to enter your Domain user name and password a second time. The Microsoft browse list may not be up to date.

It's easy to specify a drive letter mount point. Right-click My Computer and select Map Network Drive from the pop-up menu. In the Map Network Drive dialog, select from the available drive letters. Windows 2000 is a bit different from Windows NT or Windows 9x/ME in that you can configure a connection as a different user, as shown in **Figure 29**.



If you have more than one user account on a Domain, you can use this feature to connect to different shares on that Domain.

To select the desired share, enter its name in the Folder text box. Alternatively, you can view available shares. Click Browse, and then select the desired share in the Browse For Folder dialog, as shown in **Figure 30**.

Highlight a specific share from a computer on the Domain. When you click OK, the path is automatically shown in the Folder text box of the Map Network Drive dialog. If needed, you can connect with a different Domain user name and password. For example, if I logged in as user **mj**, I might want access to user **mary**'s directories. When you click the link associated with **mary**'s home directory, it opens the Connect As dialog shown in **Figure 31**.



***Figure 29**. Mapping a drive letter to a shared directory.*

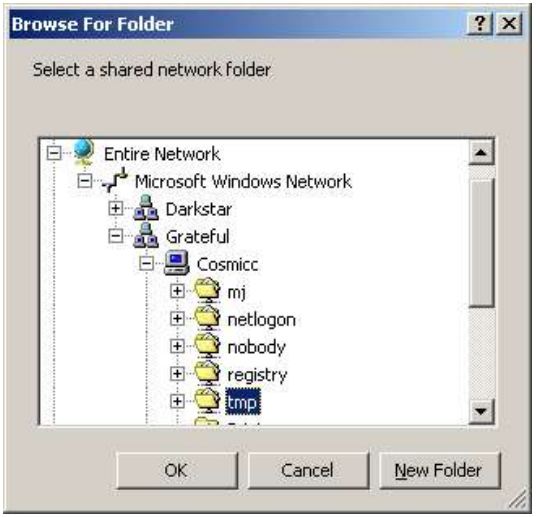


***Figure 30**. Browsing through shared directories.*



***Figure 31**. Connecting as a different Domain user.*

The shared directory is mounted to the noted drive letter, and is accessible as any other drive on your computer. Whether you can read from or copy to the share depends on the permissions set on the Linux Domain member server in the governing smb.conf file.

Finally, you can mount a network drive in a similar fashion from the MS-DOS prompt. Because the text commands apply to all Microsoft operating systems, I describe them in detail later in this chapter.

Connecting to a Workgroup share

If you've configured a Windows 2000 Professional computer on a peer-to-peer Workgroup, the process for connecting to a share depends on whether you're connecting from a Windows 9x/ME or a Windows NT/2000/XP computer. In either case, open Windows Explorer as described earlier.

For connections to a Windows 9x/ME share, you'll normally see the Enter Network Password dialog shown in **Figure 32**; you don't need to enter a user name. (Don't worry about the "Incorrect password ..." error message.) Just enter the read-only or full-access password associated with the shared directory and click OK.



***Figure 32**. Connecting to a shared directory on a peer-to-peer Workgroup.*



When you log in to a Windows 2000 Professional computer in a peer-to-peer Workgroup, the password you use matters. If it's the same password as is used to share a Windows 9x/ME directory, you won't see Figure 32. You're automatically given the Workgroup permissions (Read-only or Full) associated with your login password. Otherwise, the passwords don't match, and you'll see Figure 32 with the associated error message.

Peer-to-peer Workgroup connections to a Windows NT/2000/XP share work in the same way as from a Windows NT 4 Workstation computer. They depend on the allowed users on the target computer. If share access is limited to a specific user name, you need to log in with the same user name on the local computer.

For example, I have a share named Downloads on my Windows XP Professional computer named **allaccess**. On **allaccess**, I limit access to the user named **michael**. If I log in to my Windows 2000 Professional computer as a user named **elizabeth**, I won't have access to the \\allaccess\downloads shared directory. I need to log in as **michael** to get access to that particular share.

Windows XP Professional

In this section, I'll show you how to connect a Windows XP Professional workstation to a Domain governed by a Linux-based PDC. To this end, I'll illustrate how you can connect this 32-bit workstation to a network, set up roving profiles, connect to shared directories, and share with other computers in the network.

If you need to accommodate older Windows 95 (pre-OSR2) workstations, you can set up clear text (non-encrypted) passwords on your network. Apply the Win2000_PlainPassword.reg file described in Table 1 to your Windows XP Professional workstation registry. In this case, the same registry commands work for both Windows 2000 and Windows XP Professional.



The Windows XP Home operating system is designed only for home networks. While you can set it up on a peer-to-peer Workgroup, you can't use it to log in to a PDC. Therefore, I do not cover Windows XP Home in this book.

There are two basic steps associated with setting up a Windows 9x/ME computer on a network. First, you need to configure networking on your computer. Then you can configure a network connection to a Workgroup or Domain.

Configuring a connection to a network

To configure a Microsoft Windows XP Professional workstation on a network, have your Windows XP Professional installation CD ready, and then take the following steps:

1. Log in to Windows XP Professional, using an account with administrative privileges on the local computer.
2. Copy the following file from the /usr/share/doc/samba-2.2.7a/docs/Registry directory: WinXP_SignOrSeal.reg. It's required to allow a Windows XP Professional workstation to join a Domain configured on a Linux PDC.



*In a Microsoft Windows registry, the **RequiresSignOrSeal** value is associated with Microsoft digital signatures. The registry file disables these signatures for a connection to a Linux PDC. Also, if you're using a version of Samba other than 2.2.7a, the location of the registry file will change accordingly.*

3. Run the registry file from a viewer such as Windows Explorer; the appropriate command is automatically applied to the Windows XP Professional registry.
4. Click Start | Connect To | Show All Connections to open the Network Connections window shown in **Figure 33**. If you see a lot of connections, scroll to the bottom of the window.



***Figure 33**. Viewing Windows XP Professional network connections.*

5. Examine the icon associated with your Local Area Connection. If it's "Bridged" as shown in Figure 33, right-click the Network Bridge icon. Otherwise, right-click the Local Area Connection icon. In either case, click Properties in the pop-up menu that appears. This should open the properties associated with your connection. Click the Properties button. This should open the Local Area Connection Properties dialog shown in **Figure 34**.

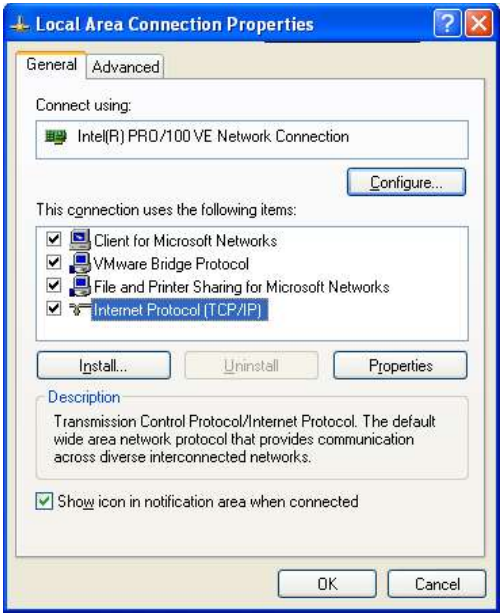


***Figure 34**. Windows XP network properties.*

6. If you don't have a DHCP server on your network, or otherwise need to manually configure your network settings, highlight Internet Protocol (TCP/IP) and then click Properties. This opens the Internet Protocol (TCP/IP) Properties window, where you can configure the IP address and DNS servers for your Windows XP Professional workstation.
7. In the Internet Protocol (TCP/IP) Properties window, click Advanced. This opens the Advanced TCP/IP Settings window, which is functionally identical to that used on a Windows 2000 Professional computer. This window is covered earlier in this chapter in Table 3.

Configuring a connection to a Domain

Now that you've set up your Windows XP Professional computer connection to a network, you can set it up for a connection to a Domain, using the following steps:

1. Click Start and point to My Computer. Right-click and select Properties from the pop-up menu that appears. This opens the System Properties dialog.
2. Select the Computer Name tab. I illustrate one view of this tab in **Figure 35**. In the "Computer description" text box, you can set the comment associated with your computer in a Domain or Workgroup browse list.

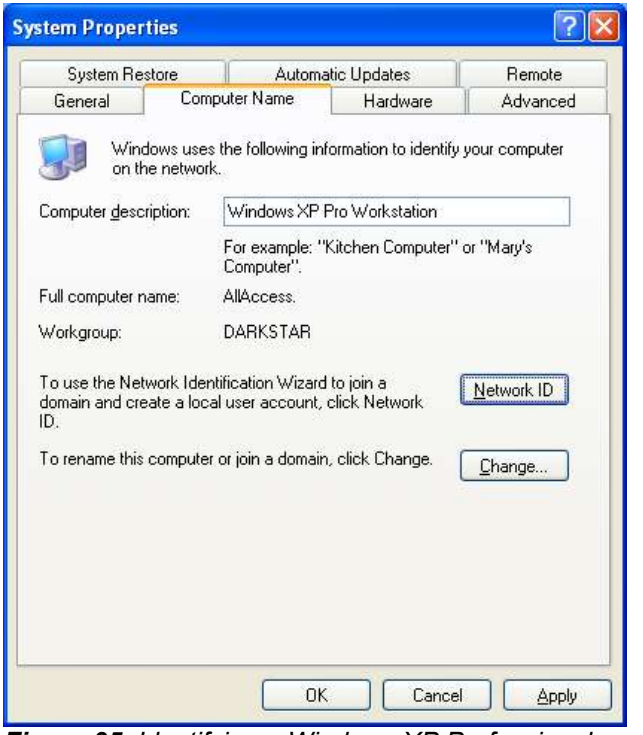


***Figure 35**. Identifying a Windows XP Professional computer.*

3. Click Change to open the Computer Name Changes dialog shown in **Figure 36**.



***Figure 36**. Configuring your computer on a Domain or Workgroup.*



Alternatively, you can click Network ID to start the Network Identification Wizard. The steps you use in this wizard for Windows XP Professional Workstation are identical to those for Windows 2000 Professional.

4. It's easy to configure this computer as part of a Domain. Click the Domain radio button, and then enter the name of the Domain that you want to join. Change the computer name if desired, and then click OK. If your computer detects a PDC on the Domain, it opens the Computer Name Changes dialog.
5. Enter the user name and password of an administrative user on the PDC. Assuming you're using a Linux PDC, enter the root user name and password.



Sometimes a workstation may not find a PDC right away, and you'll see an error. Assuming your network is properly configured, wait a few minutes and try again.

6. You should see a confirmation message such as "Welcome to the GRATEFUL Domain." Click OK.
7. You'll be told that you need to reboot before the changes take effect. Click OK, and then click OK again to exit the System Properties dialog.
8. When prompted, click OK again to reboot your computer.

Setting up a roaming profile

Roaming profiles allow users to get the same look and feel on their desktops, when they log in from any Windows NT/2000/XP workstation on a Domain. Unfortunately, Windows 9x/ME roaming profiles are not interchangeable with those available for Windows NT/2000/XP.

Not all administrators will want to create roaming profiles, because they can get quite large. For example, the roaming profile on my Windows XP Professional workstation is nearly 300MB. If your users have large roaming profiles, that can easily tax the capacity of many networks. In that case, you'll want to read the following with a view toward making sure all user profiles are local.

If you want to configure a roaming profile on a Windows XP Professional computer, follow these steps:

1. Boot your computer. When prompted, log in with an account on the Domain.



If you get an error message during the login process, you may have forgotten to apply the WinXP_SignOrSeal.reg file to your Windows XP computer registry, as described earlier.

2. Click Start. From the Start menu, right-click My Computer, and then click Properties from the pop-up menu that appears. This opens the System Properties dialog.
3. Select the Advanced tab. Under User Profiles, click Settings. Roaming profiles are enabled by default, as shown in **Figure 37**. As you can see, I've logged in as the Domain user **donna**.

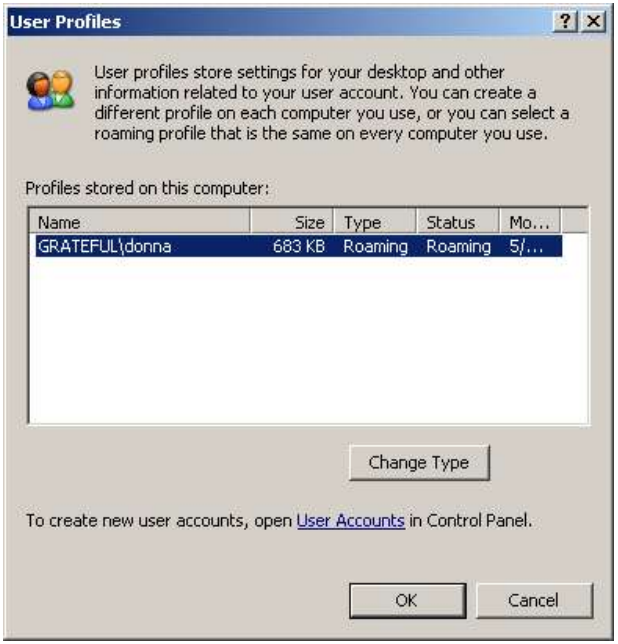


***Figure 37**. Local or roaming profile for a Domain user.*

4. Select the listed user and click Change Type. This opens the Change Profile Type dialog, which allows you to switch the configuration for this Domain user between a local and a roaming profile.

Connecting to a Domain share

The easiest way to connect to a shared directory is through My Network Places in Windows Explorer. You can also mount the shared directory to a Microsoft Windows drive letter. Later in this chapter, I'll show you how to mount to a shared directory from the command-line interface.

To access the Windows Explorer file browser, click Start | All Programs | Accessories | Windows Explorer. You can navigate to the shared directories on the Grateful Domain through My Network Places, as shown in **Figure 38**. In this case, you can see the different files available to this user from the project directory shared from the **nopaws** Linux computer. It's configured with Samba as a Windows Domain member server.

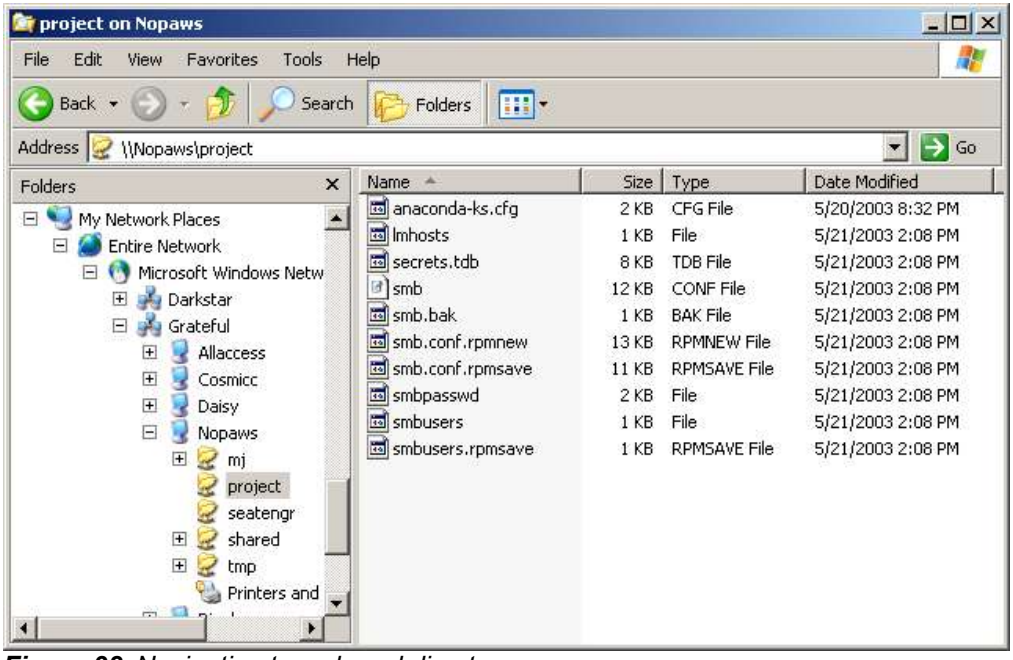


***Figure 38**. Navigating to a shared directory.*



The first time you connect to a shared directory on a Domain through a Windows workstation, you may have trouble connecting to a share where you think you have permissions. You may be prompted to enter your Domain user name and password again. That's OK; it can take a few minutes before other computers on the Domain can find your computer on the browse list.

Finally, you can mount a network drive in a similar fashion from the MS-DOS prompt. Because the text commands apply to all Microsoft operating systems, I describe them in detail later in this chapter.

Connecting to a Workgroup share

If you've configured a Windows XP Professional computer on a peer-to-peer Workgroup, the process for connecting to a share depends on whether you're connecting from a Windows 9x/ME or a Windows NT/2000/XP computer. In either case, open Windows Explorer as described earlier.

For connections to a Windows 9x/ME share, you'll see the Connect to *Computername* dialog shown in **Figure 39**; you're not allowed to enter a user name. Just enter the read-only or full-access password associated with the shared directory and click OK.



***Figure 39**. Connecting to a shared directory on a peer-to-peer Workgroup.*



When you log in to a Windows XP Professional computer in a peer-to-peer Workgroup, the password you use matters. If it's the same password as is used to share a Windows 9x/ME directory, you'll get the Workgroup permissions (Read-only or Full) associated with that password.

On a Windows NT/2000/XP share, connections work slightly differently from a Windows NT 4 Workstation/2000 Professional computer. They depend on the allowed users on the target computer. When you access the remote computer, you're prompted for the user name and password from the target Windows computer.

For example, I have a share named StStephDocs on my Windows 2000 Professional computer named **ststephen**. I limit access to the user named **michael**, as configured on **ststephen**. If I log in to my Windows XP Professional computer as a user named **donna**, Windows XP Professional prompts me for the user name and password on the Windows 2000 Professional computer.

Text-mode network commands

You can connect to shared directories from the command-line interface on a Microsoft Windows workstation. Functionally, the process is similar to the way you can map network drives in Microsoft Windows. The key commands are:

```
C:\> net view
C:\> net use
```

The **net use** command is especially useful, because it's something you can use in a logon script to connect users on your Domain to the shared directories of your choice. I'll show you how these commands work in the following sections.

Both of these commands work from the MS-DOS command-line interface, which you can open in Microsoft Windows. Click Start | Run and then enter **command** in the Open text box.

Viewing computers

Just as you can use the **smbclient** command to view the computers on a Workgroup or a Domain from the Linux command-line interface, you can use the **net view** command to view computers from the MS-DOS prompt.



*Sometimes when you run the **net** command from the DOS prompt, you may get an error message such as "Access is denied." If your connections are sound, don't let that discourage you. If you've just logged in to the computer, it may take a minute or two for the computers to find each other.*

As you can see in **Figure 40**, the **net view** command alone returns a list of the current computers in your Workgroup or Domain. The **net view /domain:darkstar** command is somewhat counter-intuitive; it allows me to view the computers in my **darkstar** peer-to-peer *Workgroup*. The command would work just as well if **darkstar** were the name of a Domain.



*If my computer is a member of a different Workgroup or Domain, I can view the computers in my Grateful Domain with the following command: **net view /domain:grateful**.*



You might note that the STSTEPHEN computer comes up in the browse list in both my Domain and Workgroup. That just tells me that I've recently moved STSTEPHEN from the Workgroup to the Domain (or vice versa).

You can also use the **net view** command to browse shared directories in any accessible computer on a Domain or Workgroup. For example, I can use the following command to view the shared directories on my Linux Domain member server named **nopaws**:

```
C:\> net view \\nopaws
```

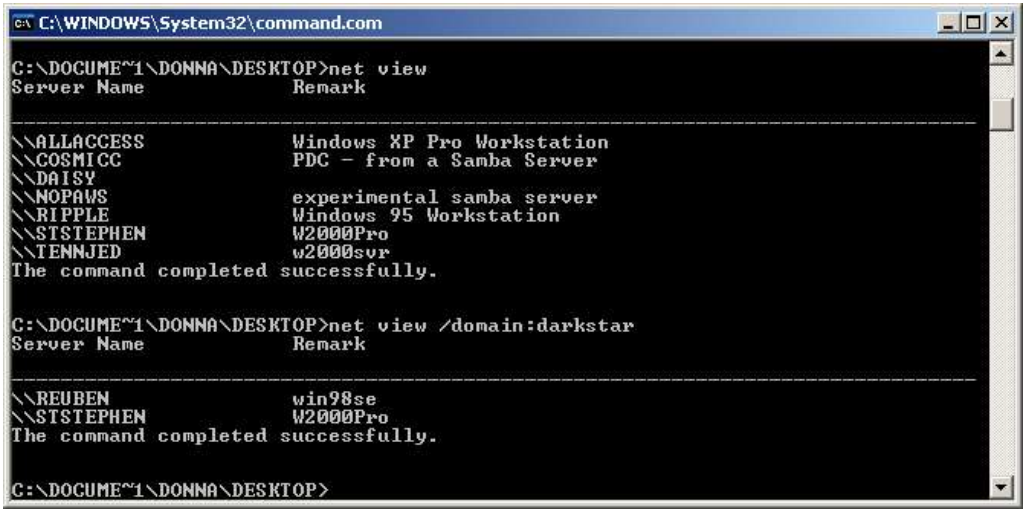


***Figure 40**. Viewing Domain and Workgroup members from the DOS prompt.*

Mounting shared directories

I've shown you the GUI way to mount a shared directory from a remote computer to a Microsoft drive letter. In Chapter 5, you learned to mount a shared directory from a remote computer to a Linux directory. You can incorporate the commands you see in this section into login batch files that run these commands automatically when a user logs in to the Domain.



As described earlier in this chapter, the batch files are located on the PDC, as defined by the [netlogon] share.

If you want to mount to a directory shared from a Domain member server, you need to log in to the workstation as the desired user. Then you should be able to mount the desired share easily with the following command:

```
C:\> net use L: \\nopaws\tmp
The command completed successfully.
```

In this case, I've mounted the [tmp] share from the **nopaws** member server on the Microsoft Windows workstation L: drive.



Sometimes on a Domain, user names and passwords don't get to the server in time; you'll need to re-enter your Domain user name and password one more time. If this happens with a login batch file, you may need to run the batch file directly from the Linux PDC's [netlogon] share, or reconnect to shared directories manually.

If you want to mount to a directory shared from a computer on a peer-to-peer Workgroup, the commands are similar. For example, I've shared the My Documents directory from my

Windows 98 computer named **reuben**. I can mount it to the M: drive with the following command (don't worry about the "password is invalid" error message):

```
C:\> net use M: \\reuben\my documents"
The password is invalid for \\reuben\my documents.

To connect reuben\Guest to reuben, press ENTER, or type a new user name:
Enter the password for 'reuben\Guest' to connect to 'reuben':
The command completed successfully.
```

The **net use** command first tries the user name and password you used to log in to the workstation. Unless there's an exact match, you'll see the "password is invalid" error message. Because **reuben** is a Windows 98 computer on a peer-to-peer Workgroup, no user name is required. Don't enter anything when prompted for a user name. When prompted for the password, enter the read-only or full-access password for that shared directory.



Observe how I've used double-quotation marks to make sure Windows reads the full name of the shared My Documents directory. Otherwise, because "My Documents" includes a space, the ***net use*** *command would look for a shared "My" directory.*

Troubleshooting

Whenever Samba starts, it's recorded in the logs. Whenever something goes wrong with Samba, it's recorded in the logs. Whenever another computer has a problem connecting to a Samba server, it's recorded in the logs. In Red Hat Linux, the Samba log files are stored in the /var/log/samba directory.

But before looking at the logs, there's the matter of syntax.

Samba syntax

It's easy to check the syntax of the Samba configuration file, smb.conf. All you need is the **testparm** command. If you have a problem with one of your variables, it should show up in the output. For example, if I misspelled something in my smb.conf file, I might see the following output:

```
# testparm -x | more
Load smb config files from /etc/samba/smb.conf
Unknown parameter encountered: "workgrou"
Ignoring unknown parameter "workgrou"
Processing section "[homes]"
Processing section "[printers]"
Processing section "[tmp]"
Processing section "[shared]"
Processing section "[project]"
Processing section "[seatengr]"
Loaded services file OK.
Press enter to see a dump of your service definitions
```

I used the **testparm -x** command to show all Samba variables that don't correspond to the default, and the **| more** switch so I can scroll through the list of variables one at a time. And

testparm identified the syntax error in my smb.conf file; I misspelled "workgroup." When you press Enter, **testparm** scrolls through the Samba variables, one page at a time.



*If you leave out the **-x** switch, **testparm** lists all variables associated with your Samba configuration. Try it out; you'll see that it's a long list.*

Samba logs

If there are problems with Samba, and the syntax of the smb.conf file checks out, the next step is to check the logs. Typical Samba problems include invalid passwords, services such as Winbind that are not running, and browser elections that are not working. Linux log files are stored in the /var/log directory; Samba log files are stored in the /var/log/samba directory. Access to these files is limited to the root user on the Linux PDC.

Take a look at these files in **Figure 41**. As you can see, it includes log files named after every computer that I've connected to this Samba server.

```
[root@CosmicC root]# cd /var/log/samba/
[root@CosmicC samba]# ls
allaccess.log    delilah.log.1    reuben.log.1    smbmount.log.2
allaccess.log.1  localhost.log    reuben.log.2    smbmount.log.3
allaccess.log.2  localhost.log.1  reuben.log.3    smbmount.log.4
allaccess.log.3  log.nmbd         reuben.log.4    ststephen.log
allaccess.log.4  log.smbd         rh9test.log     ststephen.log.1
althea.log       log.winbindd     rh9test.log.1   ststephen.log.2
cosmicc.log      nmbd.log         ripple.log      ststephen.log.3
cosmicc.log.1    nmbd.log.1       ripple.log.1    ststephen.log.4
cosmicc.log.2    nmbd.log.2       smbd.log        sugaree.log
cosmicc.log.3    nmbd.log.3       smbd.log.1      sugaree.log.1
daisy.log        nmbd.log.4       smbd.log.2      tennjed.log
daisy.log.1      nopaws.log       smbd.log.3      tennjed.log.1
daisy.log.2      nopaws.log.1     smbd.log.4      tennjed.log.2
daisy.log.3      nopaws.log.2     smbmount.log    tennjed.log.3
delilah.log      reuben.log       smbmount.log.1  tennjed.log.4
[root@CosmicC samba]#
```

***Figure 41**. Samba log files on a Linux PDC.*

Primary log files

The primary log files in /var/log/samba tell you about problems with the Samba, NetBIOS, and Winbind daemons. Here are more details about each of these log files:

- **log.smbd**: Every time the Samba daemon is started, you'll see another entry in this file, with date and time.
- **log.nmbd**: On a Linux PDC, this file includes master browser data, including elections. Every time the NetBIOS daemon, **nmbd**, is started, you'll see another entry in this file. (For more information on browser elections, see Chapter 3, "Setting Up Your Server File System.")
- **log.winbindd**: On a Linux/Samba Domain member server, this file records problems with connections to a PDC on a Microsoft Windows computer.

- **smbd.log**: Any problems with the Samba daemon are recorded in this file.



*In Red Hat Linux 9, there is a bug with the standard C language libraries, as defined by the **glibc** RPM package. This bug results in frequent error messages about "oplocks." Use the Red Hat Update Agent described in Chapter 2 to update your system. Make sure you've downloaded a **glibc** RPM package later than 2.3.2. If you don't upgrade, connection performance to a Samba PDC may be intermittent. For more information, see **bugzilla.redhat.com/bugzilla/show_bug.cgi?id=90036**.*

- **nmbd.log**: This file stores NetBIOS queries and master browser election information.

Workstation log files

For each workstation on your network, you may see two common errors:

```
Error = No route to host
Error = Connection reset by peer
```

These standard errors happen when you connect a computer to a Domain, when network connections fail, or after you restart Samba. In any of these cases, it may take some time before the PDC, master browser, and workstation computers can find each other, which leads to these noted errors. In other words, you need not worry about these errors unless the computers on your network aren't able to find each other for a reasonable period of time, such as 15 minutes. This may vary depending on your hardware setup and the number of computers on your network.

Most of the log files in my /var/log/samba directory are associated with specific workstation computers on my network. Red Hat Linux by default keeps up to five weeks of log files, rotated with a numeric extension. For example, I have five weeks of logs for connections from the computer named **ststephen**. The current week's log for connections from that computer are stored in ststephen.log. The previous week's log is stored in ststephen.log.1, and so on.

In other words, you can collect a history of logs, to check the performance of your Domain on the Linux computer that you've configured as the PDC.

Conclusion

With Samba, you can configure a Microsoft-style Domain with users on Microsoft workstations who log in to a Linux Primary Domain Controller. Before you start the process, you should check the configuration on your PDC for computer accounts, logon scripts, and user profiles as needed.

Then you can start configuring Microsoft Windows workstations to join your Domain. The steps you take for Microsoft Windows 95, 98, and ME are quite similar. While the steps you take for Windows NT, 2000, and XP are somewhat different, the basic concepts are the same:

- Make sure that networking is properly configured.
- Set up the connection to the Domain. On Microsoft Windows 9x/ME, you can do this through the network properties for the workstation. On Microsoft Windows NT/2000/XP, the steps vary.
- Once you're connected to the Domain, set up roaming profiles if desired.
- Connect to shares as needed.
- Share the directories of your choice with other users on the Domain.

You can perform many of these functions from the MS-DOS command-line interface. For example, you can browse Domains and Workgroups with the **net view** command. You can connect to shared directories with the appropriate **net use** command.

You can also set up Microsoft and Linux computers in a peer-to-peer network. When you connect to shared directories in a peer-to-peer network, the steps vary depending on whether the source is a Windows 9x/ME, Windows NT/2000/XP, or a Linux member server.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 7
Configuring Printers

Like Bogart and Bacall, Sonny and Cher, or Spike and Buffy, File and Print servers have always gone together in the world of computer networks. Just as you can configure Samba to share directories on a Microsoft Windows-style network, you can also configure Samba and a Linux print server to share printers on that same network.

Presumably, you already know how to configure and share printers from a Microsoft Windows computer. In this chapter, I'll show you how to configure and then share printers from a Linux computer. You can then connect to shared printers from either operating system from a Windows or a Linux workstation on your network.

As of this writing, Red Hat Linux only sometimes installs printers when you install the operating system. Red Hat assumes that any computer with installed printers is a print server. If you want to print from your computer or network, you may have to configure local or network print servers after you've installed Linux.

The two print servers most closely associated with Linux are the Common Unix Print System (CUPS) and the Line Print Daemon (LPD). Both are options when you install Red Hat Linux 9 on your computer. They are mutually exclusive; while you can install both services, you can run only CUPS or LPD on your network. In either case, you can use the Red Hat Printer Configurator to install local or remote printers on your Linux workstation.

CUPS also includes a Web-based interface similar to SWAT (described in Chapter 3), which allows you to configure a number of different printers as a group. This group of printers is known in CUPS as a *class*. LPD does not have its own Web-based tool. As with other Linux GUI tools, CUPS and the Red Hat Printer Configurator are both "front ends" that help you edit key text-configuration files.

Once you've configured a CUPS or an LPD printer, you can share it with other computers on your Microsoft Windows-style network through Samba. I'll show you how to connect to these printers from Microsoft Windows and Linux workstations.

This chapter provides only the briefest of introductions to CUPS and LPD. For detailed book-length information on CUPS, see the documentation available online at **www.cups.org/documentation.php**. For detailed information on LPD, refer to the Printing HOWTO, available online at **www.tldp.org/HOWTO/Printing-HOWTO/index.html**.



As of this writing, the successor to Red Hat Linux, Fedora Linux, will not include LPD software.

Packages

If you're working with Red Hat Linux 9, you may have installed the Printing Support package group described in Chapter 2, "Installing Linux as a File Server." If you installed the default packages in this group, you should have the packages you need to set up CUPS printers on your computer. Alternatively, if you upgraded from or are using an older version of Linux, you may

have the LPD system installed on your computer. For example, if you've upgraded from Red Hat Linux 7.x, LPD should still be installed on your computer.



Whether you select CUPS or LPD, make sure that you've installed the same RPMs on each of the Linux print servers and Linux workstations on your network.

In either case, Red Hat Linux lists configured printers in the /etc/printcap file. It's connected to a Microsoft Windows-style network courtesy of the **printcap name** variable in the smb.conf file on the Linux print server.

CUPS packages

As with other software in Red Hat Linux, CUPS is organized and can be installed from Red Hat Package Manager (RPM) packages. These packages are organized as files with .rpm extensions on your Red Hat Linux installation CDs. Remember, you can check to see if a package is installed by using the **rpm -q** ***packagename*** command. For example, the following command checks to see if the main CUPS server is installed:

```
$ rpm -q cups
```

I've summarized the RPM packages associated with CUPS in **Table 1**. They are installed by default if you've installed the Red Hat Linux 9 Printing Support package group. The only explicitly CUPS-related package is the **cups** RPM. The remaining packages are generic and are also required if you're using a different print service such as LPD.

Table 1*. CUPS RPM packages.*

Package	Function
4Suite	XML tools, including Universal Resource Identifier (URI) references associated with CUPS.
Xfree86-font-utils	Utilities that support printer fonts.
a2ps	Filter that converts text to PostScript format.
cups	The main service for the Common Unix Print System (CUPS).
ghostscript	An open-source implementation for PostScript.
hpijs	Optimized drivers for several types of Hewlett Packard printers.
ttfprint	Filter that converts Chinese text to PostScript format.
redhat-config-printer	The Red Hat Printer Configurator—available in Text or GUI modes.



If these packages aren't already installed, you can install them on a Red Hat Linux computer in one of two ways. You can use the Red Hat Package Management utility or the ***rpm*** *command. Both of these tools are covered in Chapter 2, "Installing Linux as a File Server."*

Several key configuration files are associated with CUPS; these are stored in the /etc/cups directory. While a full discussion of these files is beyond the scope of this book, I describe

them briefly in **Table 2**. Feel free to browse these files, before and after you finish with this chapter. You may be surprised with what you learn about printing in Linux!

***Table 2**. CUPS configuration files.*

File	Function
cupsd.conf	The basic CUPS configuration file.
classes.conf	CUPS printers can be organized into different groups, or classes, in this file.
client.conf	Sets the default CUPS server.
printers.conf	Changes from the CUPS Web-based tool are written to this file.

LPD packages

The Line Print Daemon (LPD) is also known by its RPM package name, **LPRng** (Line Print Request, next generation). It's the traditional print service for older versions of Linux and many different versions of the Unix (and clone) operating systems. Red Hat and other Linux distributions are making the transition away from LPD. There are two related optional RPMs in the Red Hat Linux Printing Support package group:

```
LPRng
redhat-switch-printer
```

redhat-switch-printer is useful if you have installed both the LPD and CUPS print services, because it allows you to switch your computer between these services. It is also known as the Red Hat Printer System Switcher.

In Red Hat Linux, there are two key configuration files associated with LPD: printcap and printcap.local, both in the /etc directory. Any changes you make with the Red Hat Printer Configurator are written to /etc/printcap; if you want to add a printer with a text editor, you'd add the information to /etc/printcap.local.

Configuring a local printer

In this section, I'll show you how to configure a printer on a local Linux server/workstation. For the moment, forget your network. Assume that you just have a local Linux PC that's directly connected to a printer. Later in this chapter, we'll repeat the process for a printer that you want to share on a network.

Checking services

Before you start installing printers, make sure CUPS is the active print service. To do so from the Red Hat Linux 9 GUI, click Main Menu | System Settings | More System Settings | Printer System Switcher. This opens the utility shown in **Figure 1**. If you've installed both CUPS and LPD, you can switch between these services. For the purposes of this chapter, make sure CUPS is selected.



Because Red Hat appears ready to remove LPD from future releases of its Linux distribution, I believe that the Printer System Switcher may also disappear from future releases.



***Figure 1**. Switching between print systems.*

When you switch print services, the Red Hat Printer System Switcher also deactivates the unwanted print service. For example, if you switch from LPD to CUPS, it stops the **lpd** service and starts the **cups** service from the /etc/init.d/rc.d directory. It also translates the code in /etc/printcap.



If you've created a printer using the Red Hat Printer Configuration utility (also known as the Printer Configurator), you can use the Printer System Switcher to switch all configured printers on your print server between the two systems.

Configuring a print service

As with Samba, there are two different GUI tools available to configure CUPS printers. One is a tool developed by Red Hat, called the Printer Configuration utility. Another is a Web-based configuration tool, developed by the people behind CUPS, Easy Software Products, Inc. (**www.easysw.com**). I'll focus on the Red Hat tool because it includes all of the settings that you'll need to configure a CUPS printer locally—or to share over a Microsoft Windows-based network.



If you want to use the default CUPS print service, don't worry about CUPS licensing. Easy Software Products has licensed the version of CUPS included with Red Hat Linux under the same license as most of the rest of the Linux operating system, the General Public License.

To start the Red Hat Printer Configuration utility from the Linux GUI, click Main Menu | System Settings | Printing. Enter the root user password if prompted; Red Hat should now open the Printer configuration dialog shown in **Figure 2**. It should also include the name of the local computer. As you can see, I'm configuring printers on my Linux Domain member server named NoPaws.



***Figure 2**. The Red Hat Printer Configuration utility.*

Now to install a local printer with the Printer Configuration tool, take the following steps:

1. On the toolbar, click New. This opens the first page of the "Add a new print queue" wizard.
2. Click Forward to open the "Queue name" page, where you'll need to enter the following:
 - A printer name such as HPLaserJ100. If you're going to share a printer on a network, you should limit the name to 13 characters; otherwise, users on Windows 9x/ME or Linux workstations won't be able to read the full name of the printer.
 - A printer description, which is used for reference within the applicable configuration files. This comment is *not* shown or added to the browse list on a Microsoft Windows-based network.
3. Enter the required information and click Forward to open the "Queue type" page shown in **Figure 3**.

 By default, the Printer Configuration tool looks at "Locally-connected" queue types. You should see available printer ports such as:
 - /dev/lp0, which corresponds to the first standard printer port on a computer. In the Microsoft world, this is known as LPT1:.
 - /dev/usb/lp0, which corresponds to the first printer connected to a USB port.
4. If you don't see your local printer port in the list, click "Rescan devices." If you have a special printer port, click "Custom device" to specify the file associated with your special printer port. This can be useful if the printers that you're planning to install haven't arrived, but you want to configure the printer on the Linux server now.
5. Select the port attached to your printer and click Forward to open the "Printer model" page shown in **Figure 4**, where you can select a printer manufacturer and model.

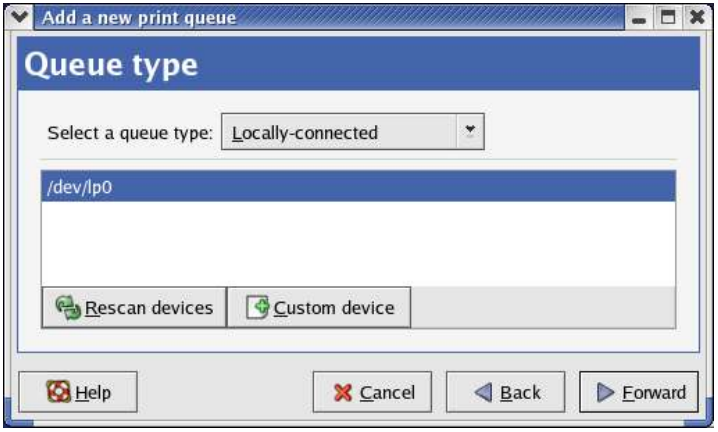


***Figure 3**. Configuring a print queue.*

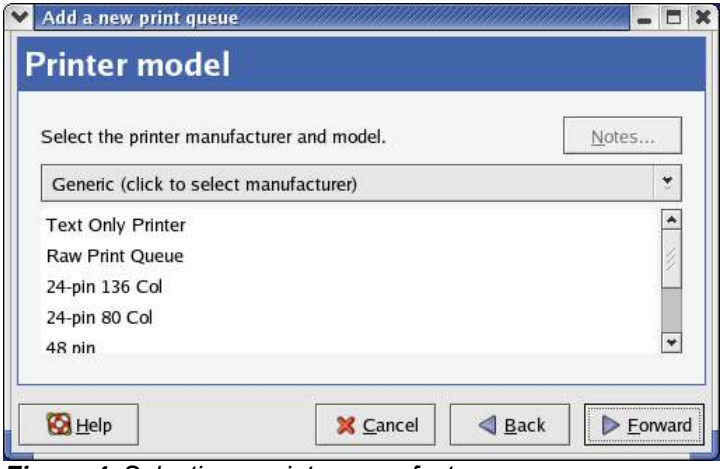


***Figure 4**. Selecting a printer manufacturer.*



You may not need to select a specific printer manufacturer or model. For example, certain Computer-Aided Design applications process print jobs without drivers. In this case, you'll want to select the "no driver" option, which is the Raw Print Queue entry shown in Figure 4.

6. This is a decision point. Are you going to install the printer locally or for the network? On a Linux computer, it's common to install the same printer twice. You might set up HPlocal for users on the local workstation and a second virtual printer, HPnetwork, for remote users. Both HPlocal and HPnetwork can refer to the same physical printer. But in this section, we're setting up a printer just for the local computer.

7. Click the "Generic (click to select manufacturer)" box to select the manufacturer of your choice. The Red Hat Printer Configurator then allows you to select from available printer models.



If you have a local PostScript printer, Linux has a special driver for you. Scroll to the bottom of the list of generic printers and then select PostScript Printers.



Once you've selected a print model, in most cases you can find more information about the selected Linux driver. Highlight the driver and click Notes. If there are limitations associated with the driver, you can read about them here.

8. Once you've selected a printer manufacturer and model, click Forward to see a summary of the selections you've made. If you're satisfied, click Apply.
9. You're asked if you want to print a test page. As long as your printer is currently connected to your computer or network, this is a good idea.
10. Now you should see your printer listed in the Printer Configuration tool from Figure 2. You can customize print settings further; highlight the printer that you just created and click Edit to open the window shown in **Figure 5**.

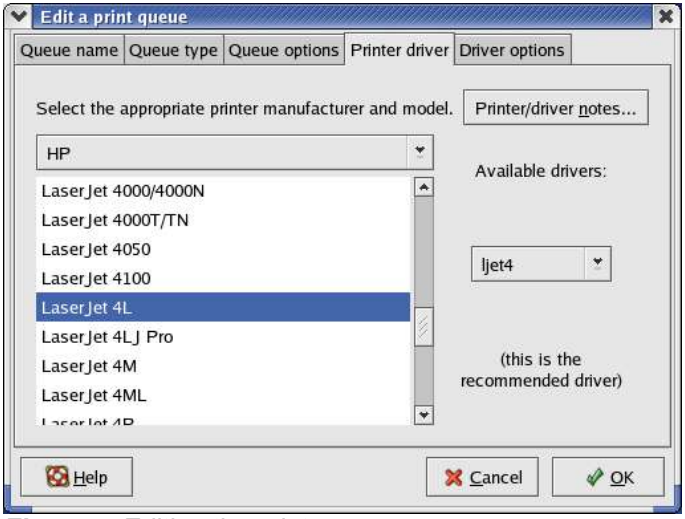


***Figure 5**. Editing the print queue.*

Table 3 summarizes what you can customize on the "Edit a print queue" tabs. If you make changes, click OK and then click the Apply button on the main Printer Configuration screen.



It appears that Red Hat may be removing the CUPS Web-based tool from future releases of Fedora Linux.

***Table 3**. Tabs in the "Edit a print queue" window.*

Tab	Purpose
Queue name	Allows you to change the name and description of the queue.
Queue type	Supports changes in the printer port; allows changes from local to networked printers. If you're not satisfied with your print driver, this tab includes an Autoselect driver option.
Queue options	Lets you customize the way the printer manages each job; options include banner pages, margins, and scaling. In most cases, the defaults are fine; non-default settings can lead to unpredictable results.
Printer driver	In many cases, more than one driver is available. Linux often includes different print drivers; one might be suitable for word processing, and another for graphics. If so, you can use this tab to switch between drivers.
Driver options	Allows you to customize print drivers. Typical options include paper source (for example, manual or a specific cassette), page size, and print quality.

Configuring printers with the CUPS Web-based tool

Like Samba, CUPS also includes a Web-based tool that you can use to configure local or network printers. It's available through the Web browser of your choice. All you need to do is navigate to **localhost:631**, which accesses the CUPS server through TCP/IP port 631. **Figure 6** illustrates the main CUPS configuration menu.

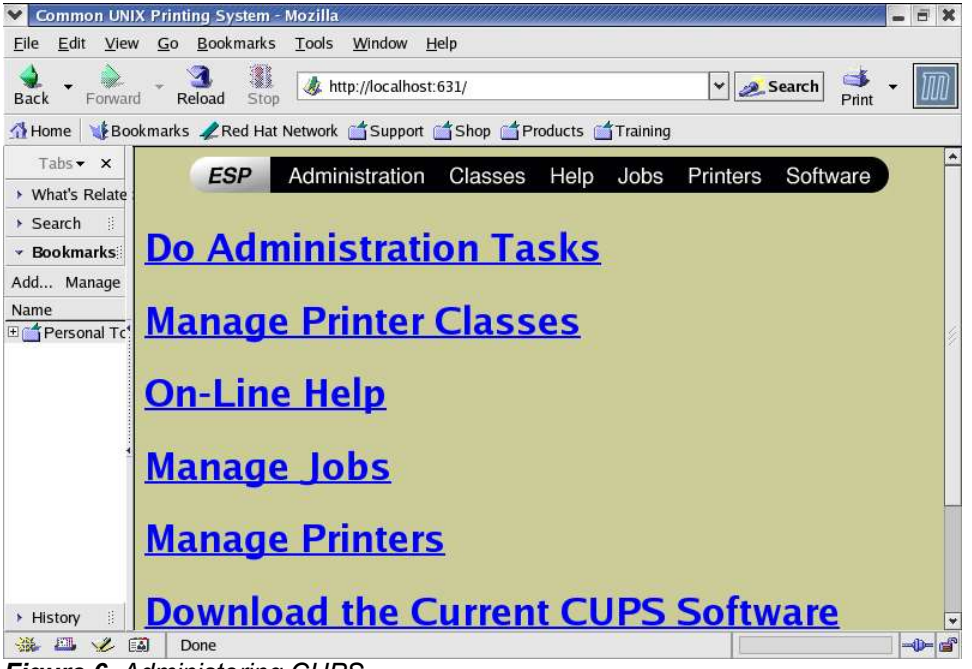


***Figure 6**. Administering CUPS.*



While any user can access the CUPS Web-based tool, administrative tasks such as adding new printers require the root administrative account and password.

With the CUPS Web-based tool, you can configure single printers locally or over a network. You can also configure a group of printers in a class. For example, if you're in a bigger organization with a group of printers in a room, you can set up a single printer *class*. You can assign all of the printers in that room to that class. When users print to that class, the print job is sent to the first available printer in that room. First, I'll show you how to access the CUPS administrative interface. Then I'll show you how to set up a printer with CUPS. Finally, I'll show you how to organize a group of printers in a class with CUPS.

Getting to the CUPS administrative interface

To access the CUPS administrative interface, take the following steps:

1. Open the Web browser of your choice. The Red Hat Linux default Web browser is Mozilla, which you can start in the Linux GUI by clicking Main Menu | Internet | Mozilla Web Browser.
2. Navigate to the CUPS Web-based tool. In the Address text box, type **localhost:631**.
3. Once the Web-based tool opens, click the Do Administration Tasks link.
4. When prompted for the user name and password for CUPS, enter **root** under the user name and the root user password. (It's required even if you're logged in to the GUI as root.) This takes you to the main CUPS administration screen shown in **Figure 7**.



***Figure 7**. The CUPS Administration screen.*

Configuring a printer

Now that you can get to the CUPS administrative interface, you can configure printers and printer classes. To install a new CUPS printer, click Add Printer. This opens the Add New Printer interface shown in **Figure 8**, where you can assign a name, location, and description to the printer.

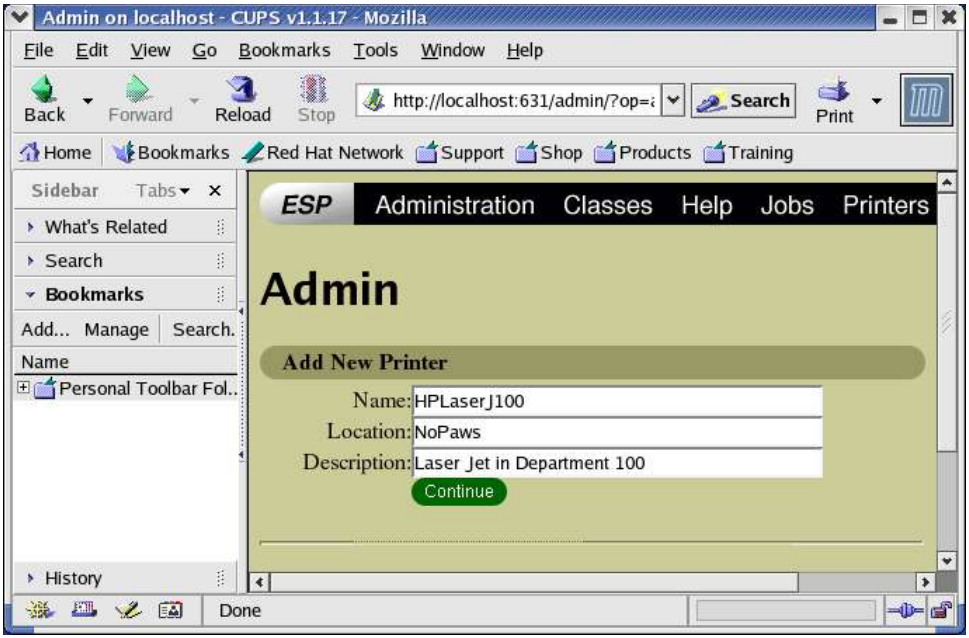


***Figure 8**. Specifying a new CUPS printer.*

Now take the following steps:

1. Enter a name for the printer. It must start with a letter and can't include spaces. If you're going to share this computer on a network, limit the name to 13 characters; otherwise, users on Windows 9x/ME and Linux workstations won't be able to read the full name of the printer.
2. Specify a location for the printer, which is the name or IP address of the computer to which the printer is connected.
3. Add the description of your choice. This comment is used to identify the printer in the applicable configuration file, /etc/cups/printers.conf. Then click Continue.
4. In the next screen, select a device for your new printer. You can select from a variety of local or network devices, including those listed in **Table 4**. Select the device associated with your printer and click Continue.

***Table 4**. CUPS printer devices.*

Device	Description
HP JetDirect	For connecting to Hewlett Packard Jet Direct network print servers.
Internet Printing Protocol	Supports print interfaces over the Internet.
LPD/LPR Host or Printer	Connects to printers configured through the other Print Service, LPD.
Parallel Port	Allows connections to local printers through a parallel port.
SCSI	Supports connections to local printers through a SCSI port.
Serial Port	For connections to printers over a serial port; allows you to customize printer port settings.
USB	Connects to printers attached to USB ports.
Windows Printer via Samba	Allows connections to printers shared via a Microsoft Windows network.



If you select a printer from a remote computer, you'll need to specify its location, known as the Universal Resource Identifier (URI). If you want more information on URIs and how you can use them with CUPS printers, click Help at the top of the CUPS Web-based tool. This leads you to the Documentation page included with the CUPS software.

5. For the purposes of this exercise, I've selected a printer connected to a standard parallel port on the local computer. Next you can select a standard printer driver. As shown in **Figure 9**, CUPS does not include the variety of drivers that are already available for the Red Hat Printer Configurator. Select the appropriate driver for your printer and click Continue.



You may not need to select a specific printer manufacturer or model. If you're setting up a printer for programs such as some CAD applications, the programs process print jobs without drivers. If you're printing from a remote Microsoft Windows computer, the Windows workstation probably already has its own print drivers. In either case, you'll want to select the "no driver" option, which is the Raw entry shown in Figure 9.



*The print models that you can configure with the CUPS Web-based tool installed with Red Hat Linux are quite limited. You can set up printers with the Red Hat Printer Configurator and then organize your printers into a class with the CUPS Web-based tool. Alternatively, you can install the CUPS software provided by Easy Software Products at **www.cups.org**. This software includes a wider variety of printers and drivers.*

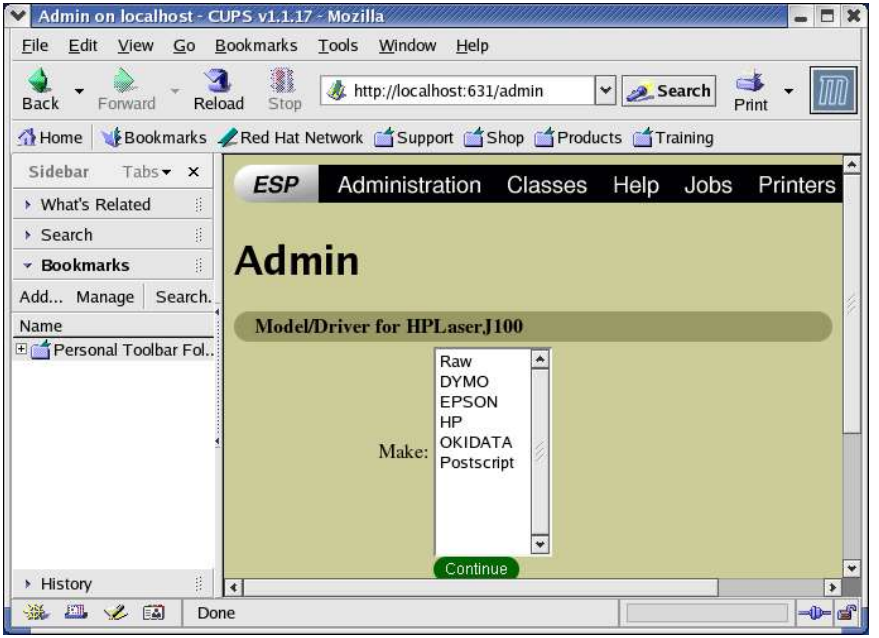


***Figure 9**. Selecting a print driver.*

6. In the next screen, select the print model most closely associated with your printer, and then click Continue.
7. If everything is okay, the CUPS tool will tell you that you've successfully added your printer.
8. Repeat the process as required for other printers on your system; or use the Red Hat Printer Configurator described earlier in this chapter.

Configuring a printer class

A CUPS printer class is a group of printers organized as a unit. Once you've set up multiple printers, you can organize the printers of your choice into a CUPS printer class. It doesn't matter whether you added the printers by using the CUPS Web-based tool or the Red Hat Printer Configurator.

Once you've configured a CUPS print class, you can print to that class as if you were printing to a single printer. CUPS then sends the print job to the first available printer in that class. To set up a print class, start the CUPS Web-based tool if it isn't already open. Navigate to the CUPS administrative interface using the steps described earlier, and then take the following steps:

1. From the CUPS Web-based tool administrative interface in Figure 7, click Add Class.
2. In the Add New Class window, enter a name, location, and description for your printer class. The same rules as adding a new printer apply. Make your entries and click Continue.

3. You'll see a list of printers configured thorough CUPS or the Red Hat Printer Configurator, similar to what's shown in **Figure 10**. Select the printers that you want to make a part of the class, and then click Continue. Use the Ctrl key if you want to select multiple printers.

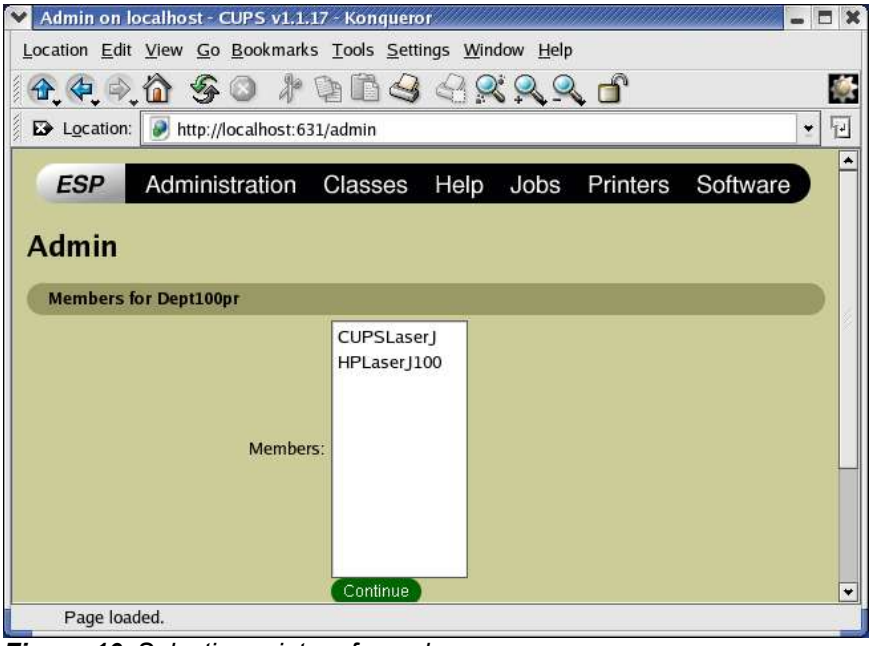


***Figure 10**. Selecting printers for a class.*

4. You should see a message that you've successfully added your new printer class.

Network printers

So far in this chapter, I've configured two CUPS printers and a print class. If you configure LPD printers, basic methods for sharing printers are the same. In both cases, the names are saved in the /etc/printcap configuration file. Here is my copy of this file, which includes the printers that I've installed so far:

```
# This file was automatically generated by cupsd(8) from the
# /etc/cups/printers.conf file.  All changes to this file
# will be lost.
WinPrint1:
WinPrint2:
WinClass1:
```

The first two entries in the list are individual printers. The third entry is a printer class. Using this configuration, I'll explain how printers are shared from a Linux computer using Samba over a network controlled by a PDC on a Windows or a Linux computer.



If you've configured LPD printers, your /etc/printcap file will look quite different. If you want to take a look at the differences, use the Printer System Switcher described earlier in this chapter to switch between CUPS and LPD. Red Hat Linux changes the /etc/printcap file when it switches between print systems.

Creating network printers

In this section, I explain how you can configure a network printer on a Linux computer. At the end of this chapter, I explain how to connect to networked printers from Microsoft Windows and Linux clients.

The local printers that I created earlier in this chapter for a Linux workstation are shared through the local smb.conf Samba configuration file. You need to do more to share these printers on a network with Microsoft Windows clients.

To understand what you need to do, first take a step back. Assume you have a Linux workstation on your Domain, connected to one physical printer. You'll want to set it up twice: The first setup is for users who print from the local workstation; the second setup allows users to connect to that printer over the network.

Earlier in this chapter I demonstrated how to set up local printers. The procedure for setting up a network printer is similar. There are two major differences:

- You should limit the printer name to 13 alphanumeric characters. Longer names don't show up in the browse list of a Windows 9x/ME or a Linux workstation.
- You should configure the printer with the Raw Print Queue printer driver. When you connect to the printer from remote computers, you'll install the driver on those computers.

The following sections briefly explain the differences between configuring a printer with the Red Hat Printer Configurator and the CUPS Web-based tool.

Creating a network printer with the Red Hat Printer Configurator

When you use the Red Hat Printer Configurator to set up a printer to share on a network, the steps are nearly identical to the steps I described earlier. When you create a network printer, keep three things in mind:

- Limit the printer name to 13 characters; **Figure 11** shows an example.
- Assuming the printer is physically connected to the local computer, select a "Locally-connected" queue type.
- When you select a printer model, select Generic and Raw Print Queue (see Figure 4). This allows remote users to set up print drivers on their local computers.

Creating a network printer with the CUPS Web-based tool

When you use the CUPS Web-based tool to set up a printer to share on a network, the steps are nearly identical to the steps I described earlier in this chapter. When you create a network printer, keep two things in mind:

- Make sure to limit the printer name to 13 characters (see Figure 8).
- When you select a printer model, select Raw Make from the options shown in Figure 9. Then you can select the Raw Queue model shown in **Figure 12**. This allows remote users to set up print drivers on their local computers.

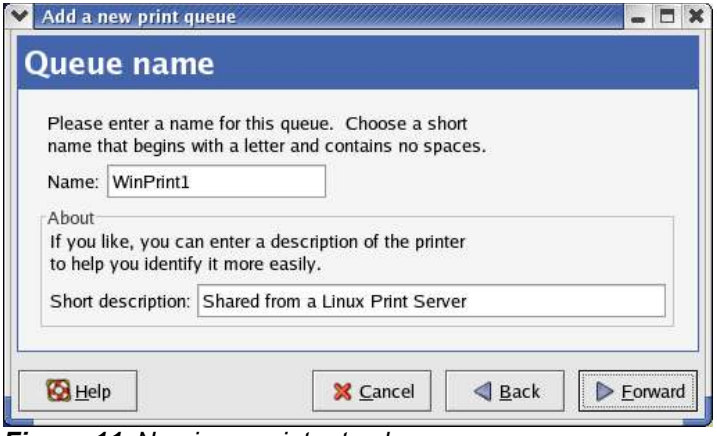


***Figure 11**. Naming a printer to share.*

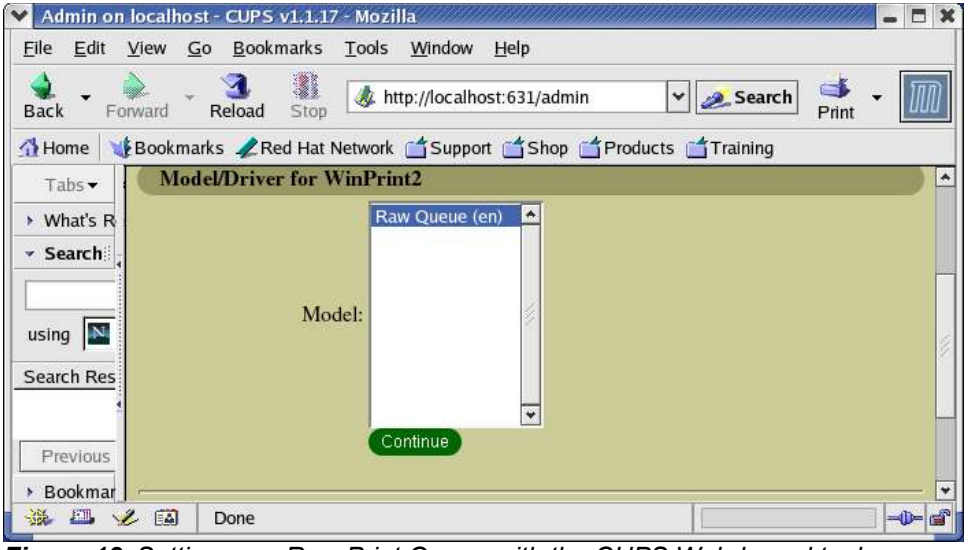


***Figure 12**. Setting up a Raw Print Queue with the CUPS Web-based tool.*

Sharing printers over a network

Now that you've created your special printers over the network, I'll show you how they're shared on a Microsoft Windows network through Samba. The key once again is the commands in the /etc/samba/smb.conf configuration file. There are two major sections in this file: [global] parameters that apply to all shares, and specific commands related to shared printers.

There are three relevant [global] commands in the default version of this file:

```
printcap name = /etc/printcap
load printers = yes
printing = cups
```

These commands are straightforward. The **printcap name** command refers to /etc/printcap for a list of printers to share. The **load printers** command loads these printers into the Windows-style networking browse list. The **printing** command refers to the print system service, typically **cups** or **lprng**. (**lprng** is another name for the LPD service.)

Once [global] commands are set, printers are normally shared in the smb.conf configuration file with the following [printers] share commands:

```
[printers]
   comment = All Printers
   path = /var/spool/samba
   browseable = no
# Set public = yes to allow user 'guest account' to print
   guest ok = no
   writable = no
   printable = yes
```

Once you've shared your printers, you can see how things look from a Windows Network Neighborhood view of shared directories and printers. **Figure 13** illustrates this view from a Windows 98 workstation on my Grateful Domain.

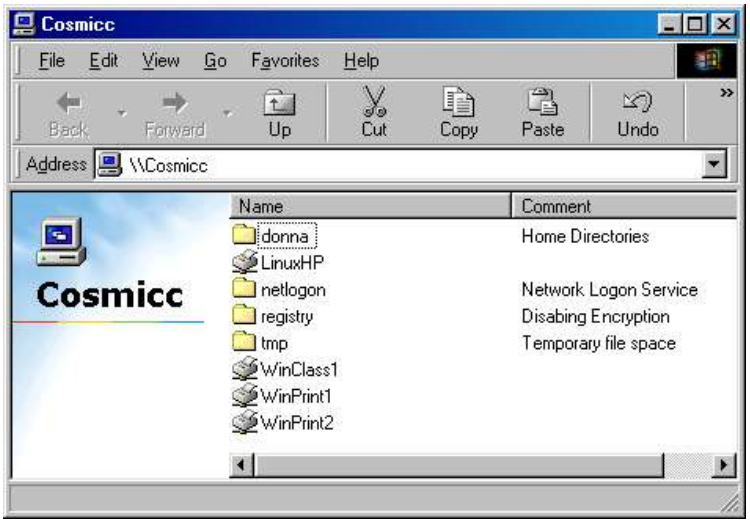


***Figure 13**. A Windows 98 workstation view of shared directories and printers.*

As you can see, the share list does not differentiate between individual printers (WinPrint1, WinPrint2) and printer classes (WinClass1) that I created. Once printers are shared through Samba, network users can print to individual printers or print classes.

Now you're ready to connect to networked printers from different workstations on your Workgroup or Domain.

Connecting to network printers from workstations

When you've configured a mixed network with Linux and Windows computers, you may have a Linux or a Windows print server. In this section, I'll show you how to connect to a printer that's been networked from either operating system. Naturally, this leads to four possible configurations:

- A Linux print server and a Windows client.
- A Windows print server and a Windows client.
- A Linux print server and a Linux client.
- A Windows print server and a Linux client.

When I refer to Microsoft Windows clients, I assume that you have access to the appropriate Windows printer drivers from your Microsoft Windows installation CD or printer manufacturer.

For the Microsoft Windows administrators among you, many of the steps in the rest of this chapter should be quite familiar.



You can set up a Linux print server with Microsoft Windows print drivers. In this configuration, you can install print drivers directly to Microsoft workstations. Details of this process are beyond the scope of this book. For more information, see ***http://us3.samba.org/samba/devel/docs/html/Samba-HOWTO-Collection.html#printing****.*

Connecting to a Linux print server from a Windows client

In this section, I assume that you have a Linux computer that is directly connected as a print server. I'll show you how to connect to a printer on that Linux print server from a Microsoft Windows client.

There are three basic ways to connect from a Microsoft Windows workstation to a Linux print server:

- A graphical connection from a remote Microsoft Windows 9x/ME workstation.
- A graphical connection from a remote Microsoft Windows NT/2000/XP workstation.
- A connection based on text commands.

Connections from a Windows 9x/ME workstation

While the steps here may seem quite familiar to Microsoft Windows administrators, this section is for the Linux administrators who may be installing a printer from Microsoft Windows for the

first time. If you're connecting from a Microsoft Windows 9x/ME workstation, take the following steps:

1. Click Start | Settings | Printers to open the Printers window.
2. Double-click the Add Printers icon to open the Add Printer Wizard.
3. Click Next. The following step in the Add Printer Wizard allows you to select between a printer attached to the local computer and a network. Because we're installing a printer from a remote Linux print server, select "Network printer."
4. Click Next. In the view shown in **Figure 14**, you can enter the direct path to the remote print queue. For example, \\cosmicc\winclass1 points to the printer named winclass1 on the Linux print server named **cosmicc**.

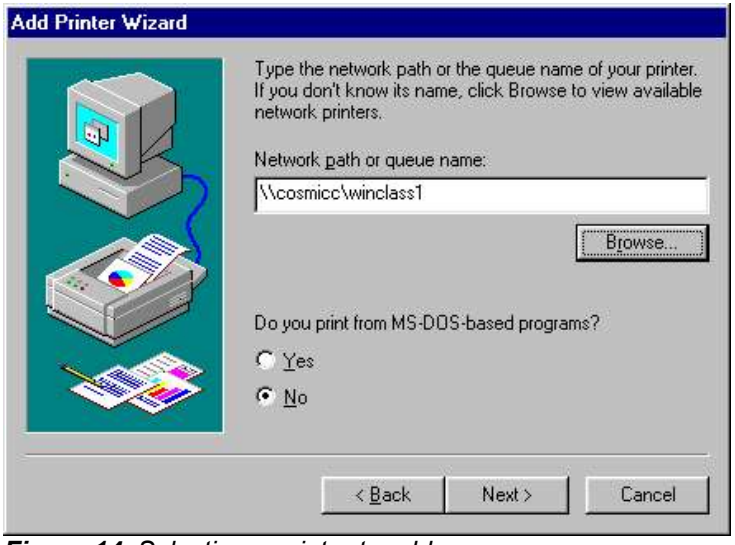


***Figure 14**. Selecting a printer to add.*



As described earlier in this chapter, a CUPS print class functions like a single printer on a browse list. The "class" is actually a group of printers. When you print to a print class, CUPS sends the print job to the first available printer in that class.

5. If you don't remember the name of the Linux print server or printer, click Browse. In the Browse for Printer window, you can select the computer and shared printer of your choice. **Figure 15** illustrates the same networked printer that I installed earlier in this chapter.
6. When you've selected the desired printer, click OK. You'll see the connection information for the printer as shown in Figure 14.

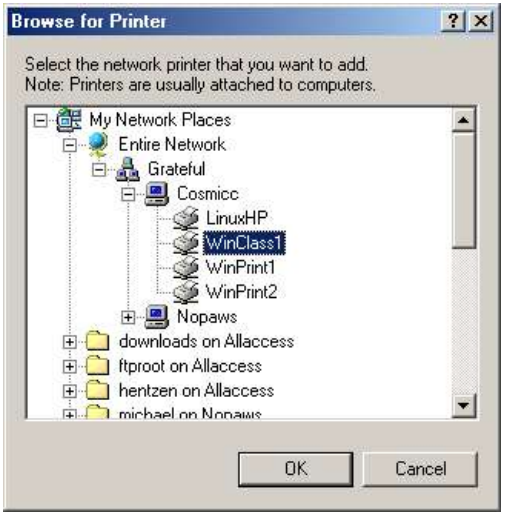


***Figure 15**. Selecting a printer from the browse list.*



The option in Figure 14 entitled "Do you print from MS-DOS-based programs?" allows you to set up this printer from pre-Windows 95 programs such as WordPerfect 5.x.

7. Click Next. In the screen shown in **Figure 16**, if you see the manufacturer and model, select them and click Next. If you're prompted for the Windows installation CD, navigate to the drive letter associated with your CD, and skip one step.



***Figure 16**. Selecting a printer.*

8. If you don't see the printer in the list, get the driver from the manufacturer. Drivers are commonly available from the associated Web site. Click Have Disk and navigate to the location of the driver files. Navigate to the location of the INF file associated with your print driver and click OK.



If you've installed the print driver before, you'll get to confirm whether you want to use the same driver.

9. Name the printer on the local computer and set it as the default. Then click Next. You now have the option to print a test page. As long as you're connected to the network, and the print server is connected to the printer, it's a good idea.
10. Click Finish. Unless you already have a direct connection to the print driver, you'll be prompted for the Windows installation CD. Insert it and click OK.



If you don't have the Windows installation CD, click OK. You're prompted for the location of the required files. If they're in a different location, you can browse and then navigate to your print drivers.

Windows now installs the print driver you selected, and prints the test page.

Connections from a Windows NT/2000/XP workstation

While the steps here may seem quite familiar to Microsoft Windows administrators, this section is for the Linux administrators who may be installing a printer from Microsoft Windows for the first time.

When you log in to the workstation, log in as a user with administrative privileges on the *local* workstation. For example, if you're logging in to the Windows NT 4 workstation named **Daisy** on the Grateful Domain, you'll need to log in as an administrative user on the local **Daisy** workstation. Administrative users on the Grateful Domain don't have privileges to install print drivers on an individual NT/2000/XP workstation.

The "look and feel" of how you install a network printer on Microsoft Windows NT 4 Workstation, Windows 2000 Professional, and Windows XP Professional are different. However, the content of most of the steps is the same. If you're connecting from a Microsoft Windows NT/2000/XP workstation, take the following steps:

1. Click Start | Settings | Printers. (In Windows XP Professional, click Start | Printers and Faxes.) This opens the Printers (and Faxes) window.
2. Double-click the Add (a) Printer icon.
3. If you're using Windows 2000 or Windows XP Professional, click Next.
4. Select whether you're installing a Local or Network printer. Because we're installing a printer from a remote Linux print server, select "Network printer server" (NT 4), "Network printer" (2000 Professional), or "A network printer or a printer attached to another computer" (XP Professional).

5. Click Next. If you're on a Windows NT 4 Workstation computer, skip the following step. If you're on a Windows 2000 Professional or a Windows XP Professional computer, you have three choices, as illustrated in **Figure 17**.
 - You can choose from available printers. (To do so on Windows 2000, don't enter any names and just click Next.)
 - You can enter the direct path to the remote printer in the Name text box shown in Figure 17. For example, the direct path to the remote print queue \\cosmicc\winprint1 points to the printer named winprint1 on the Linux print server named **cosmicc**.
 - Enter the direct URL path to the remote printer. This can work with a Linux server configured with CUPS printers.

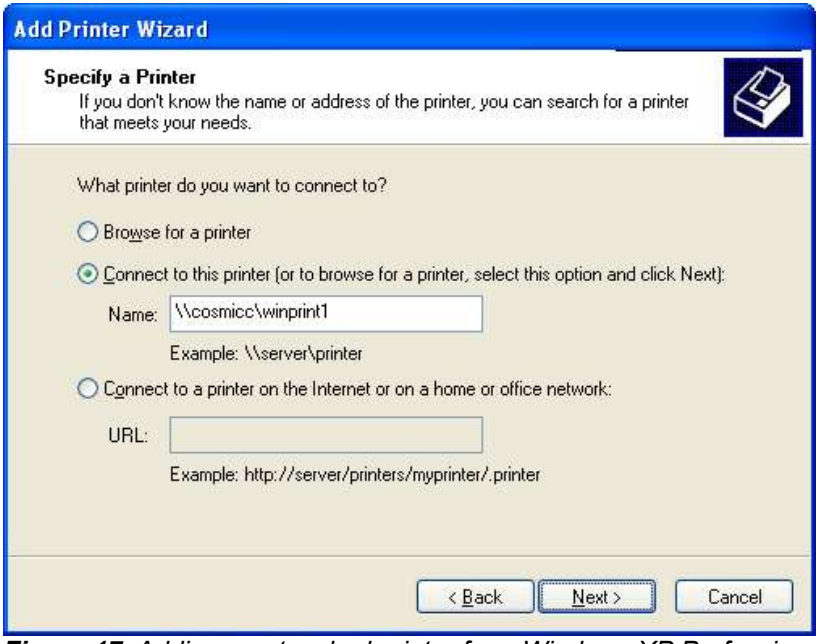


***Figure 17**. Adding a networked printer from Windows XP Professional.*

8. On Windows XP Professional, select Browse for a Printer and click Next. On Windows 2000 Professional, just click Next.

9. Now you'll be able to choose available printers from the computer of your choice. **Figure 18** illustrates the printers on the **cosmicc** computer on the Grateful Domain.

 As you can see in Figure 18, I've selected the WinClass1 printer on the computer named **cosmicc**. When I selected the printer, the Add Printer Wizard added the location in the Printer text box automatically.

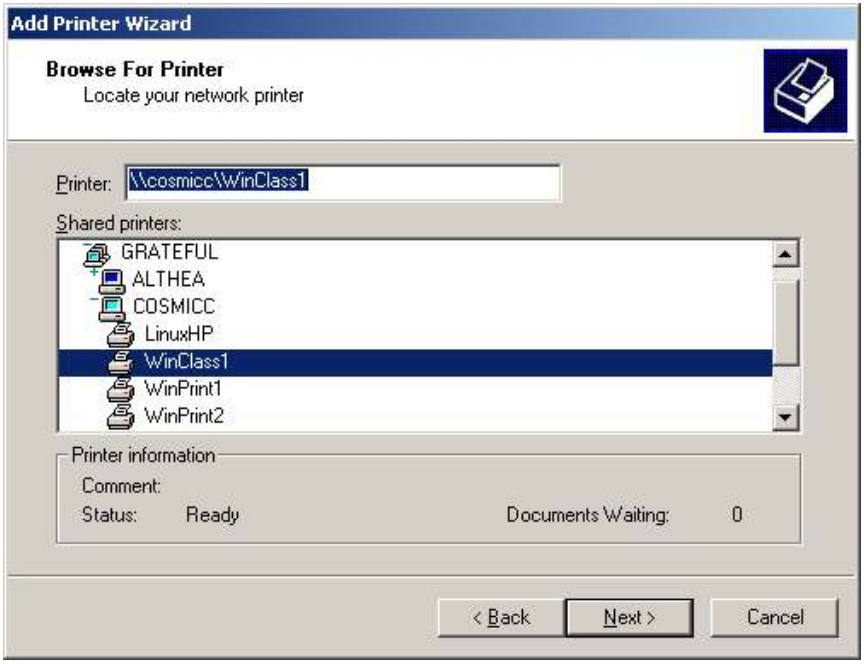


***Figure 18**. Browsing for printers from a Windows 2000 Workstation.*

11. Click Next. (On Windows NT Workstation, click OK.)



On Windows XP Professional, you'll first get a warning about possible viruses on print drivers. Assuming your PDC is on a trusted network, this should not be a concern. Click Yes when you're ready to continue.

12. Assuming that you haven't installed the Windows print drivers on the Linux print server, you'll get a warning to that effect. Click OK to accept that warning.
13. Now you'll be able to select a printer manufacturer and model, based on what you see in Figure 16. If you don't see the printer in the list, get the driver from the manufacturer. Drivers are commonly available from the associated Web site. Click Have Disk and navigate to the location of the driver files. Navigate to the location of the INF file associated with your print driver and click OK.

 If you've installed the print driver before, Windows NT/2000/XP automatically uses the same driver for your new printer.

14. Click Next. Unless you already have a direct connection to the print driver or installation CD, you'll be prompted for the Windows installation CD. Insert it and click OK.



If you don't have the Windows installation CD, click OK. You're prompted for the location of the required files. If they're in a different location, you can browse and then navigate to your print drivers.

Windows now installs the print driver you selected, and the network printer should be ready for use.

Text command connections to a network printer

If you're familiar with the MS-DOS **net use** command described in Chapter 6, you can also use it to connect network printers to a local printer port. For example, the following command connects the printer known as winclass1 on the **cosmicc** computer to the third parallel printer port:

```
C:\>net use lpt3: \\cosmicc\winclass1
```

One key difference here is that you're not prompted for a print driver. Thus, if you want to print to this port, you'll have to print from programs that don't need a separate print driver. For example, some Computer-Aided Design programs process print jobs in a form suitable for the raw output you configured earlier in this chapter.



Once you've set up the command on a Windows NT/2000/XP computer, Windows tries to reconnect to the same print port the next time you log in to that computer.

The aforementioned MS-DOS command does not add the printer to the Printers folder on Microsoft operating systems older than Windows XP Professional.



You can change any previously configured printer to take advantage of a new port. The appropriate settings are in the properties page for the printer. In the Printers folder described earlier, right-click the printer of your choice. Click Properties in the pop-up menu that appears. On Windows 9x/ME computers, click the Details tab. On Windows NT/2000/XP computers, click the Ports tab. You should easily see where you can make the desired changes.

Connecting to a Windows print server from a Windows client

It's easy to connect to a printer on a Windows print server from a Windows workstation—just as easy as connecting to a printer on a Windows print server from a Linux workstation.

The one exception is that you can install print drivers for Windows clients on a Windows NT/2000/XP print server. For example, if you're using a Windows 2000 Professional computer as a print server, you can install print drivers for Windows 9x/NT/2000 workstation computers. All you need to do is navigate to the properties page for the printer, and then select Additional Drivers on the Sharing tab. That's easy enough.

However, Microsoft does not make it easy if you have an older Microsoft operating system working as a print server. For example, it's not easy to install Windows XP Professional print drivers on a Windows 2000 Professional print server.

Because this is a book on Linux servers and workstations on a network with Microsoft computers, additional details for installing network printers from Microsoft print servers on Microsoft workstations are beyond the scope of this book.

Connecting to a Linux print server from a Linux client

In this section, I assume that you have a Linux computer on your network that is directly connected to a print server. I'll show you how to use Samba to connect to a printer on that Linux print server from a Linux client.

To connect to a printer on a remote Linux print server, you can use the Red Hat Printer Configurator described earlier in this chapter. The basic steps are similar to those I've described for configuring local printers. To connect to a remote Linux print server, take the following steps:

1. From the Red Hat Linux GUI, click Main Menu | System Settings | Printing. This opens the Red Hat Printer Configuration window.
2. Click New to open the "Add a new print queue" window.
3. Click Forward to open the "Queue name" window. Enter a name and description appropriate for the desired network printer. Remember, you should limit printer names to 13 alphanumeric characters.
4. Click Forward to open the "Queue type" window. Click the "Select a queue type" drop-down text box and then select Networked Windows (SMB). If your Linux computer is properly connected to the network, the Red Hat Printer Configurator searches for available computers.
5. Double-click the desired print server. You should see the printers that have been shared from that server though Samba. **Figure 19** illustrates my connection to the **cosmicc** computer.
6. Select the printer that you want to use, and then click Forward.
7. The Red Hat Printer Configurator takes you to the Authentication window shown in **Figure 20**. It should automatically enter the name of the print server and share. You can then make entries in the following text boxes:
 - Workgroup: Enter the name of your network Domain or peer-to-peer Workgroup.
 - User name: Enter a user name associated with the Domain or print server.
 - Password: Enter the password associated with the user name.
8. Click OK when you're done. You'll see the Printer Model window shown in Figure 4. Unless you're installing a Generic or PostScript printer, click the "Generic (click to select manufacturer)" drop-down text box.
9. From the text box, select the manufacturer for the printer. You'll see a list of printer models associated with the manufacturer. Select the appropriate model.

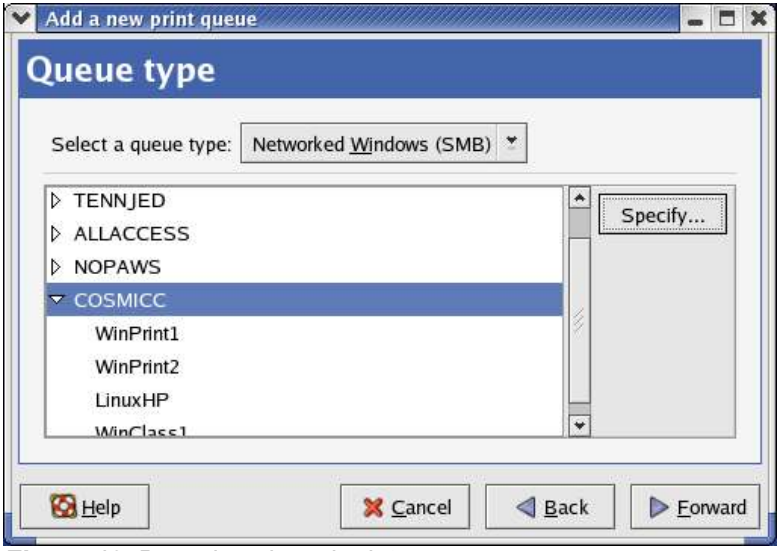


***Figure 19**. Browsing shared printers.*



***Figure 20**. Authenticating a connection to a shared printer.*

10. Click Forward to see the settings you've created. My example appears in **Figure 21**.
11. Click Apply. You're prompted to print a test page. This is an excellent idea.
12. Click Yes. The Red Hat Printer Configurator applies your changes. This may take a number of seconds. When complete, the Configurator sends a text job to your newly configured network printer.
13. Check the result from the network printer and click Yes.

 If there's a problem, you may see it in the main Red Hat Printer Configurator screen. As shown in **Figure 22**, the Configurator tells you if there is a problem making a connection.

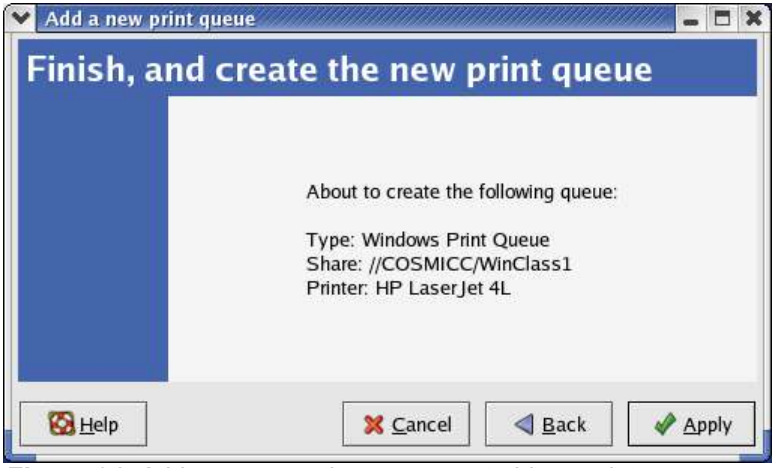


***Figure 21**. A Linux connection to a remote Linux print server.*

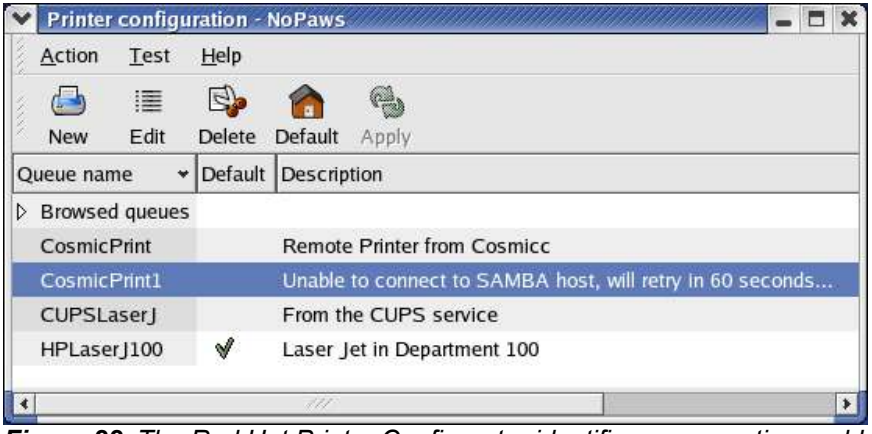


***Figure 22**. The Red Hat Printer Configurator identifies a connection problem.*

14. And that's it. You're now ready to print to the remote printer. If you've configured more than one printer on your computer, you may want to set this to be your default printer. To do so, highlight the desired printer and then click the Default button on the toolbar.

Connecting to a Windows print server from a Linux client

It's easy to connect to a printer on a Windows print server from a Windows workstation. The steps are essentially identical to those described before.

However, you might encounter one of two problems when you connect to a Windows print server from a Linux client. These can cause difficulties if you don't know how to get around them.

Some administrators set up Windows print servers with connections to other printers on remote networks. Windows administrators can share these networked printers as if they were connected directly to the Windows print server. The problem is that Samba can't read these printers in the browse list. You can still connect to these print servers, as long as you know the share name from the Windows print server.

The Red Hat Printer Configurator can't browse printers shared from Windows NT/2000 print servers. It has no problem with printers shared from a Windows XP print server. You can still find the share name of the print server by using the appropriate **smbclient** command. For example, the following output illustrates how you can view a browse list from a Windows NT 4 server named **sugaree**, which also serves as a Windows print server. This particular command prompts for **michael**'s password on the computer named **sugaree**.

```
$ smbclient -L sugaree -U michael
added interface ip=172.22.30.103 bcast=172.22.30.255 nmask=255.255.255.0
Got a positive name query response from 172.22.30.13 ( 172.22.30.13 )
Domain=[GRATEFUL] OS=[Windows NT 4.0] Server=[NT LAN Manager 4.0]
Password:

  Sharename       Type      Comment
  ---------       ----      -------
  NETLOGON        Disk      Logon server share
  Profiles        Disk      Profile Directory
  ADMIN$          Disk      Remote Admin
  HPLaserJwin     Printer   HP LaserJet 4L
  IPC$            IPC       Remote IPC
  C$              Disk      Default share
  print$          Disk      Printer Drivers

  Server               Comment
  ---------            -------
  DELILAH              Linux / Samba Workstation
  STSTEPHEN            W2000Pro
  SUGAREE              WinNT4PDC
  TENNJED              w2000svr

  Workgroup            Master
  ---------            -------
  GRATEFUL             SUGAREE
```

As you can see, the share name for the shared printer is HPLaserJwin. With the additional information in hand, you can still connect to these printers with the Red Hat Printer Configurator. It just requires a small change in procedure. The required steps follow. Because most of them are the same as those described earlier, I detail only the special steps that are required.

1. From the Red Hat GUI, click Main Menu | System Settings | Printing. Enter the root password if prompted. This opens the Red Hat Printer Configuration window.
2. Click New on the toolbar. This opens the "Add a new print queue" window.
3. Click Forward to open the "Queue name" window. Enter a name and description for the printer. Remember to limit the printer name to 13 alphanumeric characters.

4. Click Forward, click the "Select a queue type" drop-down text box, and then select Networked Windows (SMB).
5. Click Specify to open the Authentication text box. Based on the information in this section, I've filled out the settings as shown in **Figure 23** and have summarized the requirements in **Table 5**. This is different from Figure 20, as you'll need to replace the default entries in the Server and Share text boxes.



***Figure 23**. Connecting to a printer shared from a Windows print server.*

***Table 5**. Authorizing a connection to a shared printer.*

Entry	Description
Workgroup	Enter the name of the Domain or Workgroup.
Server	Enter the name of the print server computer.
Share	Enter the share name of the printer as shown in the output from the aforementioned **smbclient** command.
User name	Enter a user name on the print server or Domain.
Password	Enter the password for the given user name.

6. Click OK to return to the "Queue type" window. Be careful. No printer will be highlighted. Do not highlight any other printer.
7. Click Forward to open the "Printer model" window. Unless you're installing a Generic or PostScript printer, click the "Generic (click to select manufacturer)" drop-down text box.
8. Select the manufacturer for the printer. You'll see a list of printer models associated with the manufacturer. Select the appropriate model.
9. Click Forward to see the settings you've created.
10. Click Apply, and you'll be prompted to print a test page. This is an excellent idea.
11. After you click Yes, the Red Hat Printer Configurator applies your changes. This may take a number of seconds. When complete, the Configurator sends a test job to your newly configured network printer.
12. Check the result from the network printer and click Yes.

That's it! You're now connected from a Linux workstation to a printer that's connected to a Microsoft Windows print server.

Conclusion

Shared directories and printers have always gone together in a File and Print server. Just as you can share directories and printers from Windows servers, you can do the same from a Linux computer configured with Samba.

Red Hat Linux includes two different services that you can use to install a printer: CUPS and LPD. CUPS is the default for Red Hat Linux 9. Red Hat Linux allows you to switch between these services by using the Red Hat Printer System Switcher. Red Hat Linux includes two different GUI tools that you can use to configure a printer on the local computer: the Red Hat Printer Configurator and the CUPS Web-based tool.

If you want to install a printer on a Linux print server to share over your Domain or Workgroup, you should set up that printer without a Linux print driver. Both GUI tools include a "Raw" print queue option, which installs that printer locally without a driver. Then when you connect to that printer from a remote client, you can install the correct print driver directly on that client.

Finally, this chapter illustrated how to connect to Linux and Windows print servers from Linux and Windows client workstations on the network.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 8 Administration and Management

Linux is a sophisticated operating system with the capabilities of Windows 2003 Server and the flexibility of the Microsoft desktop operating systems. If you want to take full advantage of what Linux can do, you need to learn to administer and manage Linux from the command-line interface. While I can't turn you into an RHCE-quality Linux administrator in one chapter, I can provide some of the basic survival skills that you need to manage your Linux systems.



The Red Hat Certified Engineer (RHCE) exam is the elite Linux certification exam. It tests your mastery of Red Hat Linux in real-life situations. On this exam, you have a limited amount of time to install, configure, and troubleshoot Red Hat Linux.

In this chapter, we'll work almost entirely from the command-line interface. I'll show you how to move around and perform some basic tasks in the default Linux command-line shell, which is called *bash*. In addition, I'll give you a brief introduction to the vi editor, which may be the only command-line editor available if your system does not boot normally.

When you install Red Hat Linux on your computer, you're also installing a group of scripts that automate administrative tasks. I'll analyze two of them to help you learn to create your own scripts.

With these tools at hand, I'll then show you how to manage four key administrative jobs. First, I'll help you set up a central source for your Red Hat Linux installation files. Next, I'll show you how to work with two key files associated with the Red Hat Linux boot process, /etc/inittab and /etc/fstab. I'll guide you on the use of the Red Hat Linux rescue disk system, which can be a lifesaver if "some other" administrator accidentally tinkers too much with a key configuration file. Finally, I'll show you how to set up basic password policies for your users and synchronize password databases.

This chapter includes the most basic Linux survival skills. As you expand your Linux horizons, I recommend that you read a more general book such as *Mastering Red Hat Linux 9*, written by this author, from Sybex.



Remember, Linux is case-sensitive. For example, in Linux, the smb.conf file is different from the Smb.conf or smb.Conf files.

Shells

A shell is a command interpreter that takes what you type in at the command-line interface and passes it to the Linux operating system. Commands that you run in the shell allow you to do considerably more than any GUI tool available on a Linux or Microsoft operating system.

Commands that you type in at the shell execute faster because they don't have the "overhead" associated with the GUI.

The default shell when you install Red Hat Linux is bash, which is short for the Bourne Again SHell. It's essentially the next generation of the original command-line interpreter written by Stephen Bourne for AT&T's original versions of Unix. A number of alternative shells are available, but those details are beyond the scope of this book.



There are a number of good books on various shells and commands, such as Introduction to Unix and Linux *by John Muster, and* Teach Yourself Shell Programming in 24 Hours *by Sriranga Veeraraghavan.*

In this section, I'll show you a very few of the hundreds of basic commands that you can use in bash to navigate around your filesystem, manage the files you have, and edit key configuration files in the visual text editor, more commonly known as vi.

Basic navigational commands

Getting around the shell means knowing how to view and navigate through the files and directories on your Linux computer. There are three key commands for navigating Linux directories with the bash shell: **pwd**, **cd**, and **ls**. But before I begin, let's review the structure of Linux filesystems.

The Linux directory structure

Linux files and directories are organized in the Filesystem Hierarchy Standard, which I described in detail in Chapter 1, "Basic Linux Installation." To summarize, you can list the basic directories available under the top-level root (/) directory, as shown in **Figure 1**.

```
[mj@Delilah mj]$ ls -l /
total 193
drwxr-xr-x    2 root     root         4096 Jun  1 16:47 bin
drwxr-xr-x    4 root     root         1024 Jun  3 06:44 boot
drwxr-xr-x   20 root     root       118784 Jun 23 14:39 dev
drwxr-xr-x   56 root     root         4096 Jun 23 14:52 etc
drwxr-xr-x   10 root     root         4096 Jun  4 16:20 home
drwxr-xr-x    2 root     root         4096 Jan 24 18:52 initrd
drwxr-xr-x    9 root     root         4096 Jun  1 16:45 lib
drwx------    2 root     root        16384 May 22 15:15 lost+found
drwxr-xr-x    2 root     root         4096 Jan 27 23:22 misc
drwxr-xr-x    5 root     root         4096 May 28 17:42 mnt
drwxr-xr-x    2 root     root         4096 Jan 24 18:52 opt
dr-xr-xr-x   81 root     root            0 Jun 23 10:38 proc
drwxr-x---   15 root     root         4096 Jun 23 14:53 root
drwxr-xr-x    2 root     root         8192 Jun  1 16:47 sbin
drwxrwxrwt   18 root     root         4096 Jun 23 14:53 tmp
drwxr-xr-x   15 root     root         4096 May 22 19:17 usr
drwxr-xr-x   19 root     root         4096 May 22 19:30 var
[mj@Delilah mj]$
```

***Figure 1**. Basic Linux directories.*

To summarize the lesson from Chapter 1, here are the key directories where you'll be working with most files:

- **/boot**, with startup files such as the grub.conf boot loader and the Linux kernel.
- **/etc**, which contains most Linux configuration files such as /etc/samba/smb.conf.
- **/home**, where you can find the home directories for individual users except the administrative root user.
- **/root**, where you can find the home directory for the administrative root user.
- **/tmp**, which is the standard location for temporary files such as downloads.

I'll be taking examples from these directories in this chapter. To follow along, you'll need access to a command-line interface. If you're already in the GUI, you can open a command-line interface in one of three ways:

- Press Ctrl-Alt-F2. This opens the text-mode interface shown below. Press Ctrl-Alt-F7 to return to the GUI. By default, five text-mode terminals are available, which you can open with Ctrl-Alt-F*x*, where *x* is a number between 2 and 6. Each terminal fills up the screen.

```
Red Hat Linux release 9 (Shrike)
Kernel 2.4.20-9 on an i686

Delilah login:
```



If you've set up Red Hat Linux for a text login, six text-mode terminals are available. Linux starts in the first terminal, which you can access from other terminals by using the Alt-F1 command.

- Right-click on the desktop and then click New Terminal. You'll see a command-line window similar to Figure 1. You can add as many windows as you need in the GUI.
- Click Main Menu | System Tools | Terminal, which opens the same command-line window in the GUI.

When you log in, Linux normally takes you to the command-line interface. For example, if you log in as user **mj**, this is what you see:

```
Delilah login: mj
Password:
Last login: Mon Sep 23 14:42:53 on tty2
[mj@Delilah mj]$
```

Pay attention to the prompts. They tell you about the user who is logged in to your system. Regular users such as **mj** get the following command-line prompt:

```
$
```

In contrast, the root user administrative prompt is:

```
#
```



Some of the commands in this chapter will seem familiar. In fact, MS-DOS and Linux commands have similar roots in older versions of Unix.

pwd

The **pwd** command is straightforward; it gives you the location of your present working directory. Open a command-line interface and type the command:

```
$ pwd
/home/mj
```

The output shows that you're currently working in the /home/mj directory, which is the home directory for user **mj**.

cd

Now that you know where you are, you can navigate to the Linux directory of your choice. The standard way to move between directories in Linux is with the **cd** command. If you're familiar with MS-DOS, you'll recognize this as the "change directory" command.

To understand how **cd** works, you need to understand two concepts: absolute and relative paths. An absolute path starts with the top-level root directory, /. It doesn't matter where you start; you can run a command with an absolute path from any directory. You'll always get the same result. For example, if you want to navigate to the /etc/samba directory with the Samba configuration files using an absolute path, you'd run the following command:

```
$ cd /etc/samba
```

The relative path starts without the first forward slash. If you run a command with a relative path, the result depends on your present working directory. For example, the following series of commands work only if you have a /home/mj/etc/samba directory:

```
$ pwd
/home/mj
$ cd etc/samba
$ pwd
/home/mj/etc/samba
```

There are three important shortcuts associated with the **cd** command. For example, you can move up one level in the directory tree with the **cd ..** command. One example of how this works is as follows:

```
$ pwd
/home/mj
$ cd ..
$ pwd
/home
```



*Note that the **cd ..** command does not work in Linux unless there is a space between the **cd** and the two dots (**..**).*

If, for some reason, you don't remember the location of your home directory, the tilde (~) can help. For example, any user in the /etc/samba directory can navigate to his or her home directory by using the **cd ~** command.



*The tilde is useful for Linux scripts and programs. For example, you can set up a script that collects file names from users' home directories by using the **ls ~** command.*

ls

The **ls** command can do a lot more than just list files in a directory. With the right switches, you can list hidden files; list files with sizes and revision times; list files based on the last access time; and more.

The **ls** command by itself is simple. It lists just the non-hidden files in the current directory. For example, the following command lists the files in my home directory, /home/mj:

```
$ ls ~
book          ff05-07.tif  ff05-09.tif  ff05-11.tif  ff07-07.tif
ff05-05.tif   ff05-08.tif  ff05-10.tif  ff05-06.tif  shared
```

But you can make the **ls** command do so much more. One common variation is **ls -l**, which lists the files in the current directory in a "long listing" format. This is the version of **ls** that's most closely related to the Microsoft **dir** command. As you can see in **Figure 2**, it includes a number of other categories.

```
[mj@Delilah mj]$ ls -l
total 6380
drwxrwxr-x    1 mj       mj           4096 Jun 23 12:44 book
-rw-rw-r--    1 mj       mj         646858 Jun  2 18:33 ff05-05.tif
-rw-rw-r--    1 mj       mj         611712 Jun  3 11:22 ff05-07.tif
-rw-rw-r--    1 mj       mj         506864 Jun  3 12:28 ff05-08.tif
-rw-rw-r--    1 mj       mj         506864 Jun  3 12:43 ff05-09.tif
-rw-rw-r--    1 mj       mj        1032100 Jun  3 17:36 ff05-10.tif
-rw-rw-r--    1 mj       mj         772666 Jun  4 11:22 ff05-11.tif
-rw-rw-r--    1 mj       mj        1310756 Jun 17 09:26 ff07-06.tif
-rw-rw-r--    1 mj       mj        1090932 Jun 17 09:52 ff07-07.tif
drwxrwxr-x    2 mj       mj           4096 Jun  4 11:29 shared
[mj@Delilah mj]$
```

***Figure 2**. A long list of files in the current directory.*

For example, the key categories shown in Figure 2 correspond to the information described in **Table 1**. I use the data from the file ff05-05.tif.

***Table 1**. Long listing output.*

Item	Description
-rw-rw-r--	Permissions, described in Chapter 4.
mj (first)	User who owns the file.
mj (second)	Group that owns the file.
646858	Size of the file, in bytes.
Jun 2	Last file access date.
18:33	Last file access time.
ff05-05.tif	File name.

There are a number of other variations on the **ls** command. For example, the **ls -a** command includes hidden files in the listing. As you can see in **Figure 3**, there are a substantial number of hidden files (which start with a dot) in my home directory.

```
[mj@Delilah mj]$ ls -a
.              ff05-08.tif     .gnome             .recently-used
..             ff05-09.tif     .gnome2            .rhn-applet.conf
.bash_history  ff05-10.tif     .gnome2_private    shared
.bash_logout   ff05-11.tif     .gnome-desktop     .thumbnails
.bash_profile  ff07-06.tif     .gtkrc             .Xauthority
.bashrc        ff07-07.tif     .gtkrc-1.2-gnome2  .xsession-errors
book           .fonts.cache-1  .ICEauthority      .xvpics
.esd_auth      .gconf          .metacity
ff05-05.tif    .gconfd         .mozilla
ff05-07.tif    .gimp-1.2       .nautilus
[mj@Delilah mj]$
```

***Figure 3**. A listing of all files—including hidden files—in /home/mj.*

To find out more about the **ls** command, type the following at the command prompt:

```
$ man ls
```

This opens the manual for the **ls** command. You can scroll through this manual by using the Page Up and Page Down keys. You can exit by using the **q** command. Once you're in the manual, you can search through it using a forward slash (/). For example, if you want to find the word "sort" in this manual, type the following command:

```
/sort
```

Try it out for yourself. You'll see the **/sort** command at the bottom of the text screen.

When you browse through a Linux command manual, it may seem overwhelming; I've summarized the **ls** command switches that I use most often in **Table 2**. To learn how the **ls** command really works, I suggest that you try each of the switches for yourself.

***Table 2. ls** command examples.*

Command	Explanation
ls /etc/samba	Lists the regular files in the /etc/samba directory.
ls -a	Lists all files, including hidden files, in the current directory, in alphabetical order.
ls -F	Lists all files by type. The file type is indicated by a character at the end of the file name. A forward slash (/) means that a file is a directory, an asterisk (*) means that a file is an executable program, and an ampersand (@) means that a file is actually a link to another file.
ls -l	Lists all the files in the current directory, with output as described in Table 1. Functionally equivalent to the MS-DOS **dir** command.
ls -r	Lists files in reverse alphabetical order.
ls -t	Lists files according to the last time they were changed; by default, the most recently changed files are listed first.

You can combine command switches. For example, the **ls -ltr** command lists all files (**-l**), by the last time they were changed (**-t**), in reverse order (**-r**).

Basic file management commands

Once you know how to get around and view your files, you can poke through your filesystem. You can learn a lot about Linux by looking at the files in many key directories such as /etc/samba. But there are lots of different directories. You may need help searching through the thousands of files on your Linux system. There are two commands that can help: **find** and **locate**.

Once you find an interesting file, you may want to know what's inside. There are a variety of commands that can look through files in a number of different ways, including **cat**, **more**, **less**, **head**, and **tail**.



*You can find out more about most bash commands through their manuals; for example, to find out more about the **head** command, type **man head**.*

As you learn more about files, you'll want to know how to create and delete the files of your choice. For example, you may want to copy current files such as /etc/samba/smb.conf as a backup, or as a way to try out different configurations.

File searches

On a Linux server, users add and delete files all the time. There are two commands that you can use to find files on your computer: **find** and **locate**. There are advantages and disadvantages to each command.

find

If you want to know for sure the location of a file, you'll want to use the **find** command. It's fairly straightforward. For example, if you want to find the location of the Samba-PDC-HOWTO.html file on your system, run the following command:

```
$ find / -name Samba-PDC-HOWTO.html
```

There are two issues with the **find** command. First, if you run it as a regular user, you don't have access to, and therefore can't search through, all directories on your Linux computer. For example, the following message states that you don't have permission and therefore can't search through /root, the home directory for the administrative root user:

```
find: /root: Permission denied
```

Second, the **find** command takes time to search through all files on the computer. If a number of users are connected to the same slower computer, the **find** command could easily take several minutes.

You can use wildcards with each search. For example, if you don't know the whole file name, you can use an asterisk as a wildcard:

```
$ find / -name *Samba-*.html
```

locate

The alternative to **find** is the **locate** command. There are two advantages to **locate**. This command works very quickly, as it searches through an existing database of files on your computer. It also automatically searches for all files that have the search term anywhere in their file name. For example, the following command searches for the same files as the previous **find** command with the wildcards:

```
$ locate Samba-
```

The drawback is that the database associated with the **locate** command may not be up to date. As you'll see later in this chapter, it's updated by default at 4:02 every morning.

Reading files

Most Linux services are configured through text files. There are a number of ways to take a quick look at the files of your choice. You can use one of five basic commands to browse through a text file: **cat**, **more**, **less**, **head**, and **tail**.

cat

The **cat** ***filename*** command sends the contents of the file name to the screen. For example, the following command reads through the contents of your basic password database file:

```
$ cat /etc/passwd
```

You can also read several files contiguously; for example, the following command scrolls through /etc/passwd and /etc/group consecutively.

```
$ cat /etc/passwd /etc/group
```

more and less

The **more** and **less** commands are similar. They allow you to review a longer text file, line by line or screen by screen. The **more** ***filename*** command is straightforward; it allows you to review the contents of a file, screen by screen or line by line.

```
$ more /etc/samba/smb.conf
```

To scroll line by line, press the Enter key; to scroll screen by screen, press the spacebar key. Unfortunately, with the **more** command, you can only scroll forward in a file.

As strange as it sounds, the **less** command is more capable than **more**. When you run the following command, you can scroll up and down the length of the file.

```
$ less /etc/samba/smb.conf
```

You can use the Page Up and Page Down keys on your keyboard, as well as the arrow keys. In either case, you can search through the file as I described earlier with the **man ls** command. For example, if you're looking for the **add user script** variable in the smb.conf file, type the following after you've run the **more** or **less** command:

```
/add user script
```

To exit from the **more** or **less** scrolling interface, press **q** on your keyboard. You can also use this search method with files that are open in the vi editor.

head and tail

The **head** and **tail** commands are intuitive commands. They allow you to look at the first and last few lines in a text file.

They are essentially opposite commands. While the **head** ***filename*** command looks at the first 10 lines in *filename*, the **tail** ***filename*** command looks at the last 10 lines in *filename*. You can change the number of lines viewed by **head** or **tail**. For example, the following command allows you to look at the last 20 lines in the current NetBIOS log file:

```
# tail -n 20 /var/log/samba/nmbd.log
```



Regular users don't have permission to view Samba log files. You need to log in as the root user to view files in the /var/log/samba directory.

Creating and deleting files

To administer a Linux computer from the command-line interface, you need to know how to create and delete files and directories. Some of the commands that you will use include **touch**, **cp**, **mv**, **rm**, **mkdir**, and **rmdir**.



*If you've logged in as the administrative root user in Red Hat Linux, there are safeguards. By default, the root user can't overwrite an existing file with the **cp**, **mv**, or **rm** commands unless he or she confirms the change. For the remainder of this chapter, I'll be running commands as the root user.*

touch

In Chapter 4, I showed you how to create an empty file by using the **touch** command. For example, the following command helps you set up quotas in the /home directory:

```
# touch /home/aquota.user
```

cp

The copy (**cp**) command is nearly as versatile as **ls**. Naturally, you can copy the contents of one file to another with the following command:

```
# cp file1 file2
```

You can copy one or more files to a directory. For example, if you wanted to copy all of the files from the current directory to a directory named backup, run the following command:

```
# cp * backup
```

If a file name already exists, you're prompted to see if you want to overwrite the existing copy of the file. For example, if you run the **cp * backup** command twice, you'll be trying to overwrite the files that you just copied to the backup directory. The bash shell warns you of this problem with messages such as:

```
cp overwrite 'backup/file1'?
```

to which you can reply **y** or **n**. Alternatively, you can automatically overwrite existing files with the **-f** switch. You can search through the full **cp** command manual with the **man cp** command. Alternatively, you can look through **Table 3**, which includes examples of the **cp** command that I use most frequently.

***Table 3. cp** command examples.*

Command	Explanation
cp ***file1 file2***	Creates a new file named *file2*, and then copies the contents of *file1* into *file2*.
cp ***file1 file2 file3 dir1***	Copies the files named *file1*, *file2*, and *file3* to the *dir1* subdirectory.
cp -a ***dir1 dir2***	Copies the directory and all files within from directory *dir1* to directory *dir2*. You'll find a copy of *dir1* as a subdirectory of *dir2*.
cp -f ***file1 file2***	Copies the contents of *file1* into *file2*. If *file2* already exists, the **cp -f** command automatically overwrites it.
cp ***-i*** **file1 file2**	Copies the contents of *file1* into *file2*. If *file2* already exists, the **cp -i** command requests confirmation before overwriting it.
cp -r ***dir1 dir2***	Copies the contents of *dir1* into *dir2*. If there are subdirectories in *dir1*, those contents are also *recursively* copied to *dir2*.

mv

In Linux, when you rename a file, you're actually moving it from one file name to another. As a result, the Linux rename command is **mv**. For example, if you run the following command:

```
# mv file1 file2
```

the **mv** command creates *file2*, copies the contents of *file1* into *file2*, and then deletes *file1*. The options for the **mv** command are similar to those for the **cp** command. I illustrate some examples in **Table 4**.

***Table 4. mv** command examples.*

Command	Explanation
mv *file1 file2*	Creates a new file named *file2*, copies the contents of *file1* into *file2*, and then deletes *file1*.
mv *file1 file2 file3 dir1*	Creates new files named *file1*, *file2*, and *file3* in the *dir1* subdirectory, and then deletes *file1*, *file2*, and *file3* in the current directory.
mv -f *file1 file2*	Copies the contents of *file1* into *file2*. If *file2* already exists, the **mv -f** command automatically overwrites it. Then *file1* is deleted.
mv -*i* file1 file2	Copies the contents of *file1* into *file2*. If *file2* already exists, the **mv -i** command requests confirmation before overwriting it. Then *file1* is deleted.

rm

To delete a file in Linux, you can use the **rm** command to remove it from your system. Like any delete command on a computer, it can be dangerous if not used properly. For example, if you want to delete all files starting with *a* in the current directory, you'd run the following command:

```
# rm a*
```

Once you've deleted a file in Linux, there is no "undelete" function to restore it. For that reason, Red Hat has configured it so that the root user has to confirm before deleting a file. For example, if you ran the **rm a*** command as the root user, you'd get a message similar to:

```
rm: remove regular file 'abcd'?
```

This happens because of the Linux concept of *aliases*, where one command is substituted for another. By default, Red Hat Linux includes aliases for the root user in the standard Red Hat Linux /root/.bashrc configuration file:

```
# User specific aliases and functions
alias rm='rm -i'
alias cp='cp -i'
alias mv='mv -i'
```

If you've logged in as the root user, you may want to delete a group of files and directories with one command. For example, if you want to delete the files and directories in the /tmp/orbit-tr directory, you could run the following command:

```
# rm -rf /tmp/orbit-tr
```

The **-r** and **-f** switches work like they do for other, similar commands such as **cp** and **mv**. In other words, they force (**-f**) removal of the noted files without prompting, recursively (**-r**), which means that subdirectories of /tmp/orbit-tr are also automatically deleted.

This command can be *very dangerous*. If, as the root user, you accidentally included a space after the first forward slash, the resulting command would first delete all files on your computer, starting with the root directory; then it would try to delete all files in the /tmp/orbit-tr subdirectory:

```
# rm -rf / tmp/orbit-tr
```

> ! *Run the **rm -rf** command with extreme care. A simple mistake could delete every file on your Linux computer.*

Table 5 illustrates some safe examples for the **rm** command.

***Table 5. rm** command examples.*

Command	Explanation
rm ***file1 file2***	Deletes *file1* and *file2*.
rm ***file1 file2 file3 dir1***	Creates new files named *file1*, *file2*, and *file3* in the *dir1* subdirectory, and then deletes *file1*, *file2*, and *file3* in the current directory.
rm -f ***file1 file2***	If you've set alias **rm='rm -i'**, the **-f** forces deletion of *file1* and *file2* without prompting.
rm -r ***dir1***	Removes the contents of the *dir1* directory recursively; subdirectories and files in *dir1* are also deleted.

mkdir and rmdir

In Linux, a directory is a special type of file. You can use the **mkdir** and **rmdir** commands to create and delete these special files. In both cases, you can use the **-p** switch to create or delete multiple levels of directories. For example, the following command allows you to create directories DirA and DirB as subdirectories of the current directory, where DirB is a subdirectory of DirA:

```
$ mkdir -p DirA/DirB
```

As long as the directories are empty, you can remove both directories in the same way:

```
$ rmdir -p DirA/DirB
```

If either of these directories contains a file, the command is rejected with the following message:

```
rmdir: 'DirA/DirB/': Directory not empty
```

Adding information to the end of a file

Here's an example where you can use the **cat** command to add the previously described aliases to a user's .bashrc configuration file. Once added, the aliases can help prevent accidental file overwrites.

If you want to apply the previously described aliases to user **mj**'s home directory, run the following commands as the user **mj** or **root**:

```
# echo "# User specific aliases and functions" >> /home/mj/.bashrc
# echo "alias 'rm=rm -i'" >> /home/mj/.bashrc
# echo "alias 'rm=rm -i'" >> /home/mj/.bashrc
# echo "alias 'rm=rm -i'" >> /home/mj/.bashrc
```

The **echo** command is almost self-explanatory: It echoes the information in quotes. The double forward arrow, >>, takes the information and adds it to the end of the noted file. Be careful; if you use a single forward arrow, >, the contents within the quotes will overwrite the *entire* file.

Alternatively, if you entered the same information in a file, you could use the **cat** command to read the contents of the file to the end of **mj**'s .bashrc file. For example, if the file name is alias1, you'd run the following command:

```
# cat alias1 >> /home/mj/.bashrc
```

To make this work, you need to know how to use text editors. While you could do this in the GNOME text editor described in Chapter 3, you also need to know how to do it with a command-line text editor such as vi. If you ever have to use Red Hat's **linux rescue** mode, you won't have access to the GUI.

Text editing

As a Microsoft user and administrator, you're probably quite familiar with using editors in a GUI. Chapter 3 showed you how to use the GNOME text editor. But that's not enough, because you may need to know vi to rescue your Red Hat Linux system. The vi editor is rich with commands; this section provides the most basic of introductions.

As an example, I'll use vi to create the alias1 file described in the previous section. Because I want to append the contents of this file to the .bashrc file of a number of users, I'll do so as the administrative root user. Here's the procedure:

1. Open a command-line interface. Log in to Linux in text mode, or right-click on the GUI desktop and click New Terminal in the pop-up menu that appears.
2. At the command line, type in the **vi alias1** command. This starts the vi editor in command mode, with a prospective file name of alias1. Linux doesn't actually create the file until you save it.
3. Type the **i** command. This toggles vi into insert mode. You can then type in the desired text. In Red Hat Linux, you can use the standard keys on the keyboard to enter text and reposition the cursor.
4. When you're done entering text, press the Esc key. This returns vi to command mode.

5. To save the file, enter the **:wq** command. The colon enters execute mode, where the file is written (**w**), and then quits (**q**) from the vi editor and returns to the command-line interface.

You can then inspect the new file that you've created. Because the file is fairly short, it makes sense to list the contents with the command shown in **Figure 4**.

```
[root@NoPaws root]# cat alias1
# User specific aliases and functions
alia rm='rm -i'
aliai cp='cp -i'
alias mrv='mv -i'
[root@NoPaws root]#
```

***Figure 4**. Reading the text from a file.*

As you can see, there are some errors in this particular file. To make necessary changes, go back into the vi editor and follow these steps:

1. Run the **vi alias1** command to open the alias1 file in the vi editor.
2. Use the directional arrows on the keyboard to locate the words that you want to change.
3. You can use a variety of commands to re-enter insert mode. The **i** command is the simplest. Once you've entered insert mode, you can use the arrow keys or the Delete key to move to, insert, or delete the desired text.
4. Once you've made the desired changes, press the Esc key to return to command mode.
5. Use the **:wq** command to save your changes and exit.



*If you want to exit from the vi editor without saving changes, use the **:q!** command.*

Table 6 illustrates a number of commands that you can run in the vi editor. The best way to learn vi is to try these commands on your own Linux computer.

***Table 6**. Some vi command examples.*

Command	Explanation
a	Enters insert mode after the current position of the cursor. Contrast with the **i** command.
A	Enters insert mode at the end of the current line.
cw	Deletes the current word and enters insert mode. Contrast with the **dw** command.
dd	Deletes the current line.
dw	Deletes the current word.
Esc	Pressing the Esc key on the keyboard toggles between insert mode and command mode.
G	Moves to the last line in the file.
5G	Moves to the fifth line in the file.

Command	Explanation
i	Enters insert mode at the current position of the cursor.
o	Opens a line directly below the current position of the cursor, and then enters insert mode.
O	Opens a line directly above the current position of the cursor, and then enters insert mode.
:q	Enters execute mode and exits from vi. If you've made changes to the file, this command fails.
:q!	Enters execute mode and exits from vi, even if you've made changes to the file. All changes are lost. Contrast with the **:wq** command.
:w	Writes the current file.
:wq	Writes the current file and exits from vi.
x	Deletes a single character.
/search	Looks for the word "search" in the current file; case-sensitive.



A number of good vi reference cards are available, including one created by SSC for Linux Journal *at* ***www.ssc.com/ssc/productlist.html****.*

Scripts for repetitive tasks

Computers need to be maintained. Because maintenance affects computer and network performance, most administrators maintain networks in the middle of the night, or during periods of minimal activity. If you're administering a network of computers, you could stay up in the middle of the night with your computers, or you could set up a script to maintain them automatically.

Red Hat Linux normally installs a number of scripts that run tasks periodically. They are run by the Linux **cron** service and are stored by default in the following directories:

```
/etc/cron.hourly
/etc/cron.daily
/etc/cron.weekly
/etc/cron.monthly
```

I'll show you how scripts in these directories are scheduled, lead you through two standard daily cron scripts, and then show you how to run jobs on a one-time basis by using the **at** daemon.

The cron scheduler

By default, Red Hat Linux runs a number of maintenance jobs in the middle of the night. These jobs are run by the Linux **cron** service, as configured through the /etc/crontab configuration file. Take a look at the default version of this file in **Figure 5**.

While the first commands may seem cryptic, they are straightforward. Just as you can set *x* to a certain number in algebra, you can set variables to certain values. For example, the following command sets **bash** as the default **SHELL** variable:

```
SHELL=/bin/bash
```

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

***Figure 5**. A schedule for **cron** jobs.*

As described earlier, bash is the default shell in Linux. The next command sets a search path for commands. The **PATH** command used here tells Linux to automatically look in the noted directories for a given command in the order they are listed:

```
PATH=/sbin:/bin:/usr/sbin:/usr/bin
```

For example, if you run the **run-parts** command, Linux automatically searches the /sbin, the /bin, the /usr/sbin, and then the /usr/bin directories for that command, in that order. It runs the first **run-parts** command that it finds in one of these directories.

The **MAILTO** command is useful for administrators; any time Linux runs a **cron** job, this command sends a message to the noted user. The following command sends the message to the Linux administrative user, root. You can set this to the user name of your choice. If you've set up a mail server, you can even set this to your Internet e-mail address.

```
MAILTO=root
```

Finally, the **HOME** variable sets the starting directory for the remaining commands in this script—in this case, the top-level root (/) directory:

```
HOME=/
```

As described in earlier chapters, any command that starts with a pound sign (#) or a semicolon (;) is a comment; thus the **#run-parts** line is just a comment for the commands that follow. Now things become a bit more complex. In the /etc/crontab file, there is a specific format for a command. For example, the first command from the default version of this file runs every script in the /etc/cron.hourly directory at one minute past every hour of every day:

```
01 * * * * root run-parts /etc/cron.hourly
```

The next command runs the scripts in the /etc/cron.daily directory at 4:02 a.m. daily:

```
02 4 * * * root run-parts /etc/cron.daily
```

The third command runs every script in the /etc/cron.weekly directory at 4:22 a.m. every Sunday:

```
22 4 * * 0 root run-parts /etc/cron.weekly
```

The final command runs every script in the /etc/cron.monthly directory at 4:42 a.m. on the first day of each month:

```
42 4 1 * * root run-parts /etc/cron.monthly
```

As you can see, there is a definite pattern to each of these commands. The first five columns determine the time that **cron** runs the command. The administrative user, root, runs the **run-parts** command with the data as shown. In each case, the input to the **run-parts** command is a directory. **Table 7** details the meaning of each of these columns, from left to right.

***Table 7**. /etc/crontab command columns.*

Item	Description
Minute	Set this column between 0 and 59. If you substitute an asterisk, the command is run every minute.
Hour	Set this column between 0 and 23, where 0 represents 12 midnight and 12 represents 12 noon. If you substitute an asterisk, the command is run every hour.
Day	Set this to the desired day, between 1 and 31. If you substitute an asterisk, the command is run every day.
Month	Set this to the desired month, where 1 represents January and 12 represents December. If you substitute an asterisk, the command is run every month.
Day of week	Set this to the desired day, where 0 represents Sunday and 6 represents Saturday. If you substitute an asterisk, the command is run every day of the week.

With Table 7 in mind, you can reschedule standard jobs or set up new jobs. One thing to keep in mind is that you can set a range; for example, if you set 1-5 in the third column, the job is run every day from Monday through Friday.

For example, if you configure an /etc/cron.yearly directory, you could set it so **cron** runs the scripts in that directory at 30 minutes past midnight on January 1:

```
30 0 1 1 * root run-parts /etc/cron.yearly
```

I've kept the asterisk on the fifth column, because the day of the week associated with January 1 changes from year to year.

cron rotates logs

Chapter 6 mentioned that Samba log files are rotated on a weekly basis. That's made possible through the logrotate script in the /etc/cron.daily directory. Based on the standard /etc/crontab file, Linux runs logrotate at 4:02 a.m. every day. It's a simple file, with two lines. The first line might be a bit confusing:

```
#!/bin/sh
```

I did say that every file in a script that starts with a pound sign (#) is a comment. Well, the "#!" makes it an exception to the rule. The first line in a script, when written in this way, can

determine the command interpreter used in the rest of the script. In that way, it works like the **SHELL** command described earlier.

And the **/bin/sh** command represents the default Linux bash shell, just as the **SHELL=/bin/bash** command does in the /etc/crontab script. The second line in the logrotate script is also simple. It runs the **logrotate** command from the /usr/sbin directory with the /etc/logrotate.conf configuration file:

```
/usr/sbin/logrotate /etc/logrotate.conf
```

cron helps find files

I described the **locate** command earlier in this chapter as one that finds files quickly because it searches through an existing database of files. Well, by default, Red Hat Linux regenerates that database daily, courtesy of the slocate.cron script in the /etc/cron.daily directory.

This script is a bit more complex; it includes three lines. You'll recognize the first line from the slocate.cron script as one that sets this script to use the bash shell:

```
#!/bin/sh
```

The second line looks complex; the intent is simple. The **renice +19** command makes sure that this script doesn't run if there's almost anything else running on your system. It does this by lowering the priority of the threads associated with the script. Explaining the details of this command is well beyond the scope of this book.

```
renice +19 -p $$ >>/dev/null 2>&1
```

The final line also looks complex. But it applies the **updatedb** command from the /usr/bin directory. This command searches through every file on your system and reads it into the database associated with the **locate** command. The **-f** switch makes sure that **updatedb** does not look through directories that are formatted to the noted filesystems. The **-e** switch excludes specific directories from the search. The following command is actually on one line:

```
/usr/bin/updatedb -f "nfs,smbfs,ncpfs,proc,devpts"
-e "/tmp,/var/tmp,/usr/tmp,/afs,/net"
```

As described in Chapter 1, Linux directories are normally formatted to Linux's third extended filesystem, ext3. The previous command excludes directories mounted to other filesystems, including the Network File System (NFS), which is used to share files between Linux and Unix computers. Directories mounted with the **smbmount** command are automatically mounted with the Samba filesystem, **smbfs**. The previous command also excludes a number of other filesystems, which I do not cover in this book.

The previous command also excludes (**-e**) a number of directories, such as those which normally store only temporary files: /tmp, /var/tmp, and /usr/tmp.

I normally add one more directory to this list: /mnt. If you've inserted a floppy or a CD in a drive while in the Linux GUI, the default Red Hat GNOME configuration utility automatically mounts this temporary media. CDs especially contain a lot of files. Unless you always keep the same CD in the drive, it's a good idea to exclude the files on the CD from your

locate command file database. Otherwise, the next time Linux runs the slocate.cron script, it will catalog the files on any mounted CDs.

The change is simple; I add **/mnt** to the end of the command in the slocate.cron script:

```
/usr/bin/updatedb -f "nfs,smbfs,ncpfs,proc,devpts"
-e "/tmp,/var/tmp,/usr/tmp,/afs,/net,/mnt"
```

A one-time job

What if you have a large database that takes a lot of computer resources that you just wanted to run over the weekend? You could sacrifice your weekend and run the job personally. You could set up a **cron** job; unfortunately, if you forget to deactivate the job, the **cron** daemon might rerun the job a number of times.

In Linux, administrators run one-time jobs through the **at** daemon. The **at** command actually initiates a shell-like interface where you can enter the jobs of your choice. Assume that you want to process the /home/mj/galley database file through the /usr/sbin/kitchen script starting this Saturday at 1:00 a.m. If today is Wednesday, you'd run the following commands:

```
$ at 1:00 + 3 days
warning: commands will be executed using (in order) a) $SHELL b) login shell
c) /bin/sh
at> /usr/sbin/kitchen /home/mj/galley
at> <Ctrl> D
```

The warning specifies the shell. By default, the value of **$SHELL** is **/bin/bash**. You can check this with the **echo $SHELL** command. As described earlier in this book, the default login shell is bash. And as noted earlier in this chapter, **/bin/sh** also starts the bash shell.

The previous command sets up the noted job to be run in three days at 1:00 a.m. You can check this by using the following command:

```
$ atq
2      2003-06-28 01:00
```

This command tells me that **at** job number 2 is to be run on December 20, 2003 at 1:00 a.m. You can check out how **at** will execute this job in the /var/spool/at directory. The applicable job is owned by the desired user. You can inspect this with the following command (as the root user):

```
# ls -l /var/spool/at
total 12
-rwx------     1 root     root        1794 Jun 25 14:27 a00001010cc08c
-rwx------     1 mj       mj          1309 Jun 25 14:53 a00002010cc08c
drwx------     2 daemon   daemon      4096 Jan 24 16:45 spool
#
```

In this case, the **at** jobs for user **mj** are stored in the a00002010cc08c file. If you want, you can delete job 2 with the following command:

```
$ atrm 2
```

Administrative tools

As a Linux administrator, you have a number of other key tools at your disposal. The tools covered in this section are not difficult to use, but can help you in so many ways. While I've shown you how to mount the Red Hat Linux installation CDs, it would be easier if you could access the Red Hat Linux installation files from a single source, over a network.

To paraphrase what you might experience as an administrator, corruption happens. Critical files may be lost, or another Linux administrator could make a mistake. To understand where things might go wrong, I'll show you the basics of the Linux boot process. Then I'll show you how to use the Linux rescue disk. Finally, I'll help you learn how to manage your users in more detail.

Managing the installation files

Chapter 2 showed how to install and remove RPM packages with the **rpm** command. But that command is not enough for a Linux system administrator. There are three Red Hat Linux installation CDs, and you may not always have the CDs with you.

Creating an installation source

Once you've copied the Red Hat Linux installation files to a single source on your network, you can connect to it like any other shared Samba directory. To copy the Red Hat installation files, first create a shared RedHat directory, copy the base and RPMS subdirectories from the first installation CD, and then copy the RPM package files from the second and third installation CDs. I'm assuming that you already know how to do this on a Microsoft computer. In either case, you'll need a computer with about 2GB of free space. To set this on on a Linux computer, follow these steps:

1. Log in as the root user. If you're in the GUI, open a command-line interface. Right-click on the desktop, and click New Terminal in the pop-up menu that appears.
2. Create a separate directory for installation files. For the purpose of this exercise, I'll create the /mnt/source directory.

   ```
   # mkdir /mnt/source
   ```

3. Mount the first Red Hat Linux installation CD. If you're in the GUI, that should happen automatically when you insert the CD; otherwise run the following command:

   ```
   # mount /mnt/cdrom
   ```

4. Copy the required files and directories from the first CD. Remember, because you're transferring several hundred MB of files, this may take a few minutes. As described in Table 3, the following command copies all files within the RedHat directory to /mnt/source, recursively; when complete, you'll find the RedHat directory and all subdirectories under /mnt/source.

   ```
   # cp -ar /mnt/cdrom/RedHat /mnt/source
   ```

5. Unmount the first CD and insert the second installation CD. If you're not in the GUI, mount it as described in step 3. Copy the files from the second CD.

```
# umount /mnt/cdrom
# mount /mnt/cdrom
# cp -ar /mnt/cdrom/RedHat /mnt/source
```

6. Repeat step 5 with the third installation CD.
7. Make sure to share the /mnt/source directory. In the Samba smb.conf configuration file, you'll need to insert a stanza of commands similar to the following:

```
[inst]
      comment = Red Hat Linux 9 CD files
      path = /mnt/source
      browseable = yes
      read only = yes
```

8. Save your changes to the smb.conf file. Remember to make Samba reread your changes with the following command:

```
# /sbin/service smb reload
```



You can even install Red Hat Linux over the network—if you share it through an NFS, FTP, or Web server. For more information, see the Sybex book Mastering Red Hat Linux 9*, written by this author.*

Using the installation source

Now you can mount this new source of installation files from other Linux computers on your network. The basic steps are the same whether you've set up the Red Hat Linux installation files on a Windows or a Linux computer. Assume the computer is named **cosmicc**, and your user name on the domain is **tr**. To mount the Samba [inst] share on the /home/tr/source directory, log in to the local computer as user **tr**, and then run the following **smbmount** command:

```
# smbmount //cosmicc/inst /home/tr/source
```

Assuming user **tr** also exists on the PDC, this **smbmount** command automatically mounts the [inst] share from the **cosmicc** computer on the /home/tr/source directory.



*If you see an error message when you try the **smbmount** command as a regular user, you may need to apply the **chmod u+s** command on the **smbmnt** and **smbumount** commands. For more information, see Chapter 5, "Connecting Linux Workstations."*

After you've mounted the Red Hat Linux installation files on a single directory, you have a single source from which to install the RPMs from the Red Hat Linux CDs.

Managing the Linux boot process

Whenever you boot a Red Hat Linux computer, it runs a set of commands that checks your hardware, loads the kernel, mounts standard directories, sets basic initialization parameters, and then loads the services you need for a fully functional Linux server or workstation. As described in Chapter 3, you can configure different services to start if you set your computer to log in at the text or GUI interfaces.

While this may seem like a lot of trivia, you need to know your configuration files to this level of detail if you ever need to rescue your Linux system. One of the biggest problems with these files are typos from administrators who don't know the details. I'll cover three of the key configuration files associated with the Linux boot process:

- /boot/grub/grub.conf is the configuration file associated with the default GRUB boot loader.
- /etc/inittab sets the default runlevel and the standard text-mode login screens.
- /etc/fstab mounts and sets up preconfigured directories with filesystems.

Pay attention to these files. Whenever I've had a problem booting Linux, I've found that the cause is most often related to one or more of these files. For more information, see the Red Hat documentation on this topic, available at **www.redhat.com/docs/manuals/linux/**.

The GRUB boot loader

Once configured, GRUB allows you to load the operating system or the Linux kernel of your choice. Because it's such a key file in the boot process, it's worth analyzing in some detail. Let's take a careful look at the grub.conf configuration file on my desktop computer, which illustrates a "dual-boot" configuration. In other words, the grub.conf file shown in **Figure 6** allows you boot into Linux or Microsoft Windows.



There are two major boot loaders associated with Linux. GRUB is the Grand Unified Bootloader, which is the current default for Red Hat Linux. LILO is the Linux Loader, which was the default for Red Hat Linux through early 2002. If you want to convert from LILO to GRUB, read the GRUB manual. You can read it on most Linux computers with the ***info grub*** *command.*

Now let's analyze the GRUB commands in detail. The first line indicates that this version of grub.conf was first created using Anaconda, which is Red Hat's code name for the installation program that was covered in Chapters 1 and 2:

```
# grub.conf generated by anaconda
```

The next comment indicates that any changes you make to GRUB are automatically read the next time you boot your computer.

```
# Note that you do not have to rerun grub after making changes to this file
```

```
# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,1)
#          kernel /vmlinuz-version ro root=/dev/hda3
#          initrd /initrd-version.img
#boot=/dev/hda
default=0
timeout=10
splashimage=(hd0,1)/grub/splash.xpm.gz
title Red Hat Linux (2.4.20-9)
        root (hd0,1)
        kernel /vmlinuz-2.4.20-9 ro root=LABEL=/ hdc=ide-scsi
        initrd /initrd-2.4.20-9.img
title Microsoft Windows Server
        rootnoverify (hd0,0)
        chainloader +1
```

***Figure 6**. GRUB configured for a dual-boot.*

Unlike other boot loaders (for example, LILO or Windows' NTLDR), you don't have to write boot-loader changes to any hard drive master boot record (MBR).

As you may remember from Chapter 1, I installed Red Hat Linux with the /boot directory mounted on a separate partition. The following notice indicates that the Linux kernel (vmlinuz) and Initial RAM disk (initrd) files are both stored in the /boot directory.

```
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,1)
#          kernel /vmlinuz-version ro root=/dev/hda3
#          initrd /initrd-version.img
```

Now this is where it's easy to get confused. The **root (hd0,1)** comment tells me that the /boot directory is located on the first hard drive (0), on the second primary partition (1). But the **root=/dev/hda3** comment tells me that the top-level directory, which is also called the "root directory," is located on the third primary partition of the first IDE hard drive.



*It's easy to get lost with Linux's different definitions of root. There's the root user, which is the administrative user on the Linux computer. The root user's home directory is /root. The /root directory is just one subdirectory. The top-level directory in Linux, /, is also known as the root directory. And in the GRUB configuration file, the **root(hd0,1)** variable represents the partition with the /boot directory.*

Finally, the following comment tells me that the Linux /boot directory is located on the first IDE hard drive on my computer:

```
#boot=/dev/hda
```

If the /boot directory were located on the first SCSI drive, this line would read:

```
#boot=/dev/sda
```

We've just looked at the comments. Now let's look at the actual commands in my grub.conf configuration file. The first command refers to the default operating system. In other words, the following command sets Red Hat Linux as the default:

```
default=0
```

If I changed this variable to **default=1** in this file, that would point to the second stanza, which would set Microsoft Windows Server as the default. The next command sets a timeout; in this case, if I don't run a command or click the up or down arrow within 10 seconds, GRUB automatically boots the default operating system.

```
timeout=10
```

Red Hat Linux has its own standard graphical image that goes with the GRUB boot loader, which is associated with the **splashimage** command. Based on the **root(hd0,1)** comment earlier in this file, the splash.xpm.gz file is located in the /boot/grub directory.

```
splashimage=(hd0,1)/grub/splash.xpm.gz
```

You'll see how this looks shortly. But first, let's look at the first stanza. The **title** command sets the label that you'll see in the GRUB menu:

```
title Red Hat Linux (2.4.20-9)
```

The next line tells GRUB the location of the /boot directory. As described earlier, it's on the second primary partition on the first IDE hard drive:

```
root (hd0,1)
```

The following line actually represents *three* different commands. The first part of the command tells me that GRUB can find the Linux kernel in the /boot directory, in the vmlinuz-2.4.20-9 file, and that it should be read into the operating system as a read-only (**ro**) file. The next part of the command labels the top-level Linux directory, /, as root. The final part of the command is a message passed to the kernel about hdc, which typically represents the CD Drive on a PC. The idc-scsi module is required by Linux to read some CD drives and is normally automatically included for required drives during the Red Hat Linux installation process.

```
kernel /vmlinuz-2.4.20-9 ro root=LABEL=/ hdc=ide-scsi
```

When Red Hat Linux first boots on your system, it starts by loading an Initial RAM disk, which then loads other drivers from Linux to your computer. The following line locates the Initial RAM disk on the /boot directory.

```
initrd /initrd-2.4.20-9.img
```

Now we'll look at the other stanza, which boots some version of a Microsoft Windows Server operating system on this computer. As before, the **title** command sets the label that you'll see in the GRUB menu:

```
title Microsoft Windows Server
```

The next command locates the boot files associated with this operating system. The **rootnoverify** command means that GRUB doesn't look for a Linux /boot directory in the Windows partition. The **(hd0,0)** label means that the Windows boot files are located on the first partition of that first hard drive.

```
rootnoverify (hd0,0)
```

Finally, this last command, for a Microsoft Windows server, loads the NT Boot Loader from the first sector of the noted partition. You can inspect the result of this configuration file in **Figure 7**.



***Figure 7**. The GRUB boot screen.*

Starting Linux with /etc/inittab

Once you've selected Linux, the boot process eventually moves to the /etc/inittab file. This file determines whether Linux starts in text or GUI mode, starts a list of services in that mode, configures standard text login screens, and sets a few standard commands. You may want to change one of these commands for the sake of your Windows users.

As with grub.conf, the first part of /etc/inittab includes comments that help you understand the purpose and some basic parameters for this file. As you can see, there are seven possible runlevels available for Red Hat Linux. The only runlevels that you normally need to consider are 3 (Full multiuser mode) and 5 (X11). As discussed in Chapter 2, these runlevels correspond to text and GUI logins.

```
# Default runlevel. The runlevels used by RHS are:
#   0 - halt (Do NOT set initdefault to this)
#   1 - Single user mode
#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)
#   3 - Full multiuser mode
#   4 - unused
#   5 - X11
#   6 - reboot (Do NOT set initdefault to this)
```

The first command sets the default runlevel. The number 5 in this command tells me that, by default, this computer starts in a GUI login screen. If you prefer a text login screen, you can change the 5 to a 3 in the text editor of your choice.

```
id:5:initdefault:
```

The following command completes the basic boot process, which runs the rc.sysinit script, located in the /etc/rc.d directory:

```
si::sysinit:/etc/rc.d/rc.sysinit
```

Linux uses only one of the next seven commands, the one that corresponds to the default runlevel. For example, if the default runlevel is 5, the applicable command starts the scripts in the /etc/rc.d/rc5.d/ directory.

```
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
l3:3:wait:/etc/rc.d/rc 3
l4:4:wait:/etc/rc.d/rc 4
l5:5:wait:/etc/rc.d/rc 5
l6:6:wait:/etc/rc.d/rc 6
```

The following command allows any user to press Ctrl-Alt-Del to reboot this Linux computer. If you're going to have users who are more familiar with Microsoft Windows, they might press Ctrl-Alt-Del at the first sign of trouble. Unfortunately, that reboots this Linux computer for *all* users. I'd disable this command by adding a pound sign (#) in front of the **ca** command line.

```
# Trap CTRL-ALT-DELETE
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

The following commands relate to the installation of an uninterruptible power supply on this computer. You can read more about installing a UPS on your system at **www.tldp.org/HOWTO/UPS-HOWTO.html**.

```
# When our UPS tells us power has failed, assume we have a few minutes
# of power left.  Schedule a shutdown for 2 minutes from now.
# This does, of course, assume you have powerd installed and your
# UPS connected and working correctly.
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"
# If power was restored before the shutdown kicked in, cancel it.
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"
```

Even if you've set up Linux to log in at the GUI, the following commands configure six text login terminals. You can switch between these terminals with the Ctrl-Alt-F*x* command, where *x* is a number between 1 and 6. A text-mode login terminal is known as a *getty*.

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

If you've configured the GUI, the first text login terminal does not work. Finally, there's one last command, which does not apply unless you disable Red Hat's default GNOME GUI login terminal:

```
x:5:respawn:/etc/X11/prefdm -nodaemon
```

Standard mounts with /etc/fstab

When Linux boots on your computer, it relies on /etc/fstab for a list of directories to mount on different partitions. These are known as filesystems; thus, this is the filesystem (**fs**) tabulation file. **Figure 8** illustrates one example from my Red Hat Linux computer; the **LABEL** command is used for the standard directories that are mounted when your computer boots Linux.

```
LABEL=/                 /                       ext3    defaults        1 1
LABEL=/boot             /boot                   ext3    defaults        1 2
LABEL=/home         /home           ext3  defaults,usrquota,grpquota     1 2
none                    /dev/pts                devpts  gid=5,mode=620  0 0
none                    /proc                   proc    defaults        0 0
none                    /dev/shm                tmpfs   defaults        0 0
/dev/sda3               swap                    swap    defaults        0 0
/dev/fd0                /mnt/floppy             auto    noauto,owner,kudzu 0 0
/dev/cdrom              /mnt/cdrom              udf,iso9660 noauto,owner,kudzu,ro 0 0
```

Figure 8*. Standard mounts from /etc/fstab.*

Unfortunately, this file is more complex than it looks, but a detailed description is beyond the scope of this book. I'll explain some basics about the first three and the last two lines in this particular file. The first three lines are associated with actual directories mounted on my hard drives:

```
LABEL=/                 /               ext3    defaults                        1 1
LABEL=/boot             /boot           ext3    defaults                        1 2
LABEL=/home             /home           ext3    defaults,usrquota,grpquota      1 2
```

Note the third line, which was modified in Chapter 4 to work with user and group quotas. The last two lines relate to drives that you can install after Linux starts: a floppy and a CD:

```
/dev/fd0                /mnt/floppy     auto            noauto,owner,kudzu      0 0
/dev/cdrom              /mnt/cdrom      udf,iso9660     noauto,owner,kudzu,ro   0 0
```

You can see how this works through the **df** (disk free) command shown in **Figure 9**. It illustrates the partitions as filesystems mounted on specific directories.

```
[root@NoPaws root]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda2              3644800   2838492    621160  83% /
/dev/sda1               101089     19571     76299  21% /boot
/dev/sdb2              1009912    657572    301040  69% /home
none                     62992         0     62992   0% /dev/shm
/dev/fd0                  1430      1413        18  99% /mnt/floppy
/dev/cdrom              653312    653312         0 100% /mnt/cdrom
[root@NoPaws root]#
```

***Figure 9**. Filesystems mounted on directories.*

For example, the /dev/sda1 filesystem represents the first primary partition on the first SCSI hard drive. It works with the second line in /etc/fstab as long as this command works:

```
# e2label /dev/sda1
/boot
#
```

Table 8 shows how each entry in /etc/fstab works.

***Table 8**. /etc/fstab command columns.*

Column	Item	Description
1	LABEL	The **LABEL** variable such as /boot or the partition such as /dev/cdrom to be mounted.
2	Mount Point	The directory where the partition defined by the **LABEL** variable is to be mounted.
3	Format	The format type; *ext3* is the Red Hat Linux default; *vfat* corresponds to FAT16 or FAT32; *auto* checks the format of the media; *udf,iso9660* corresponds to the standard format for data CDs.
4	Mount Options	The **defaults** mount option works with a partition that is mounted automatically, read-write, when you boot Linux. There's more to this option, which is beyond the scope of this book.
5	Dump value	If 1, files are automatically written to disk. If 0 (such as for a floppy drive), you should run the **sync** command to write a file to disk.
6	Filesystem check order	Linux periodically checks filesystems for integrity with the **fsck** command, which is similar to the Microsoft **chkdsk** command. The root directory (/) requires a value of 1; all other mounted directories require a value of 2. Linux doesn't normally perform a **fsck** on floppy and CD drives.

The Red Hat Linux rescue mode

If you ever have a problem booting Red Hat Linux on your computer, you can use the boot floppy disk that you created during the installation process. You can still create that boot floppy after Red Hat Linux is installed. For example, if you see a message like the following, you can insert the rescue floppy and boot normally into your Linux system.

```
  Booting 'Red Hat Linux (2.4.20-9)'

root(hd1,0)
 Filesystem type unknown, partition type 0x83
kernel /vmlinuz-2.4.20-9 ro root=LABEL=/

Error 17: Cannot mount selected partition

Press any key to continue...
```

If you press a key, you're taken back to the GRUB menu. That's not good enough if you have only one version of Linux on this computer. But you can the fix the problem. If you've read the previous sections, you'll know that the problem is with the **root** variable in the GRUB configuration file, /boot/grub/grub.conf. You can start your system by booting your computer from the rescue floppy. You can then edit and fix /boot/grub/grub.conf or restore it from a backup.

If you've forgotten to create or have lost that boot floppy, you'll have a little more trouble booting into your system. But all is not lost. You can rescue many failed Red Hat Linux systems by booting from the first Red Hat Linux installation CD, in what is known as **linux rescue** mode.

Rescue disks

You should have created a rescue floppy during the installation process, as described in Chapter 2, "Installing Linux as a File Server." But there are two problems with rescue floppies. They are easy to lose, and they are customized for a specific Linux kernel. For example, the kernel that's installed with Red Hat Linux 9 is version 2.4.20-8.

Red Hat has already released later versions of its customized Linux kernel. You can upgrade to the latest Red Hat kernel with the Red Hat Update Agent described in Chapter 2. If you do, you'll also need to upgrade your rescue disk. For example, if you upgrade to a hypothetical kernel version 2.4.21-2, you can create a new rescue floppy. Insert a 1.44MB floppy into the appropriate drive and enter the following command:

```
# mkbootdisk 2.4.21-2
```

When you reboot your computer from a rescue disk, you'll see a message similar to the following:

```
SYSLINUX 2.00 2002-10-25 Copyright (C) 1994-2002 H. Peter Anvin

Press <return> (or wait 10 seconds) to boot your Red Hat Linux system from
/dev/sda2. You may override the default linux kernel parameters by typing
"linux <params>", followed by <return> if you like.

boot:
```

Linux rescue mode

If you've lost your rescue disk, all is not lost. You can boot into most broken Red Hat Linux systems by using the first Red Hat Linux installation CD. No, you're not reinstalling Linux. You're just rescuing a system that you could not otherwise boot.

The Red Hat Linux rescue mode is powerful. Recently, I accidentally deleted the master boot record from my desktop computer. I had installed Windows 98 and Red Hat Linux 9 in a dual-boot configuration. I tried various Windows 98 boot floppies. They did not even recognize the hard drive. The **linux rescue** mode that I explain in this section helped me fix the hard drive quickly and easily.



*The steps in this section are based on Red Hat Linux 9. If you're using another version of Red Hat Linux, the steps may vary slightly. The **linux rescue** mode did not exist before Red Hat Linux 7.3.*

To use **linux rescue** mode, take the following steps:

1. Boot your computer. When it first boots, enter the BIOS menu. Depending on your computer, you can do this by pressing the F1, F2, or Delete keys. If none of these keys work, check the documentation for your computer.
2. Adjust the boot order of your computer, so it boots first from your CD drive.
3. Insert the first Red Hat Linux installation CD, and then reboot your computer.



If you have a Red Hat Linux installation boot floppy, you can substitute it for the first Red Hat Linux installation CD.

4. When you see the Red Hat Linux installation screen, you'll see the following at the bottom of the screen, where you can type the **linux rescue** command.

```
[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]
boot: linux rescue
```

5. The Red Hat Linux installation program installs a basic Linux system on your computer. You'll need to go through a few basic installation steps before **linux rescue** mode searches through your system. In the Choose a Language screen, select a language and then press Enter.
6. In the Keyboard Type screen, select the keyboard that most closely reflects your system, and then press Enter.
7. Your CD starts running Anaconda. When you see the Setup Networking screen, use the Tab key to highlight No, and then press Enter.
8. You'll see the Rescue screen shown in **Figure 10**. You have three choices here; the Read-Only and Skip options require skills that are beyond the scope of this book. Therefore, highlight Continue and then press Enter.

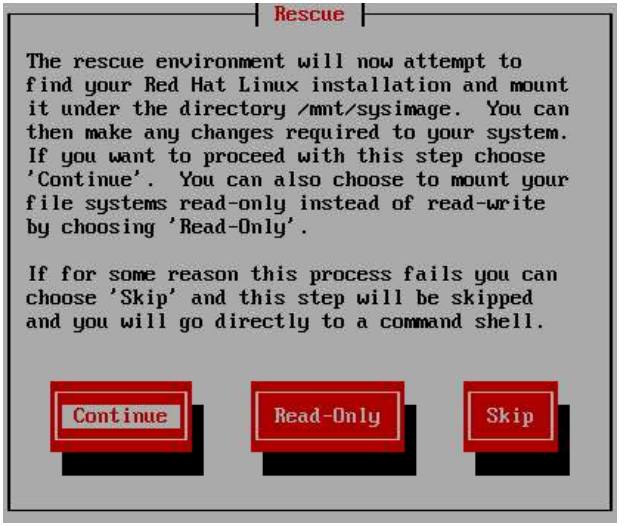


***Figure 10**. The Rescue screen.*

9. Red Hat's **linux rescue** mode looks through the partitions on your hard drive(s) for the proper files in your /boot directory. If it finds them, you'll see the screen shown in **Figure 11**.

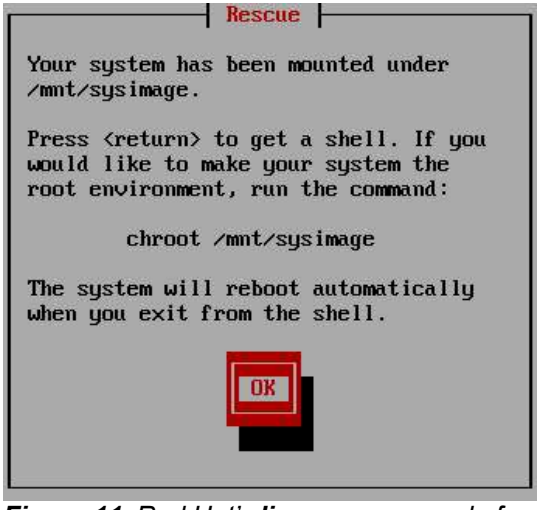


***Figure 11**. Red Hat's **linux rescue** mode found your Linux installation.*

10. Press OK to see the **linux rescue** command prompt, as shown:

```
Your system is mounted under the /mnt/sysimage directory.

When finished, please exit from the shell and your system will reboot.

sh-2.05b#
```

11. For now, the top-level root directory (/) is actually mounted as the /mnt/sysimage directory. As suggested in Figure 11, you can restore your Linux directory tree with the following command:

```
sh-2.05b# chroot /mnt/sysimage
```

12. Now you can edit or replace the files of your choice. For the problem described at the beginning of this section, you'll want to replace the GRUB configuration file with a backup. If you don't have one available, open the /boot/grub/grub.conf file with the following command:

```
# vi /boot/grub/grub.conf
```

13. Make the changes required in grub.conf and save. Type the **reboot** command to reboot your computer.
14. Remove the CD from its drive. If you've made the right changes, Linux should now boot normally.

Regular user tasks

There are two more straightforward tasks that you should know about as a Linux administrator. First, you should know how to set up password policies for your users. Second, you should know how to synchronize password databases, in the rare case where you need to replace a Linux PDC. You can set password policies for specific users with the Red Hat User Manager. You can set this up in general through the /etc/login.defs configuration file.

Password policies

To set up a password policy for a specific user, you can use text commands such as **chage**, or you can use the Red Hat User Manager described in Chapter 4, "Setting Up Your File Server's Users." The Red Hat User Manager is fairly capable. To start it from your Linux GUI, click Main Menu | System Settings | Users and Groups.

Now select the user of your choice. To configure the password policy for that user, click the Properties button. This opens the User Properties window, where you can click the Account Info tab. Using this tab, you can:

- Set an expiration date for the account.
- Lock the account by deleting the password from /etc/shadow. If you reactivate the account, you'll need to set the password again.

You can also manage password policies for a specific user by using the Password Info tab shown in **Figure 12**. There are five things you can set under this tab. I list a few variables, which I'll explain later in the section.

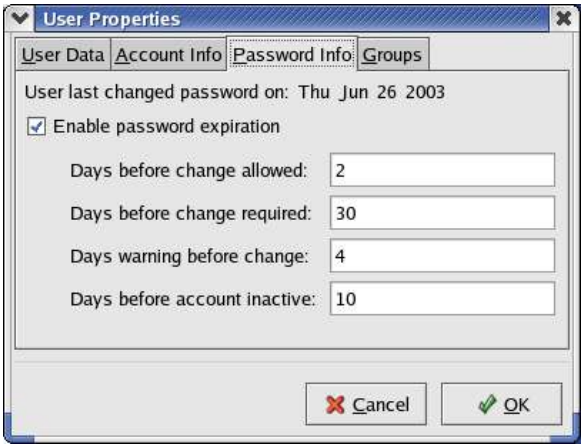


***Figure 12**. Setting the password policy for a user.*

- A password for a specific user does not expire unless you select the "Enable password expiration" check box.
- You can force a user to keep a password for one or more days in the "Days before change allowed" text box. The user associated with Figure 12 can't change her password until June 28. This corresponds to the **PASS_MIN_DAYS** variable.
- You can force a user to change a password periodically; this user has to change his password by July 26. This corresponds to the **PASS_MAX_DAYS** variable.
- This Linux computer warns the user that she needs to change her password by July 22. This corresponds to the **PASS_WARN_AGE** variable.
- Finally, if this user doesn't change her password, the account is made inactive 10 days after July 26.

You can check these settings for user **waymon** from the command-line interface by using the **chage -l waymon** command. But you're an administrator. If you're responsible for a large number of users, you don't want to have to set policies one at a time. You can set a default password policy for all new users through the /etc/login.defs configuration file.

There are four key variables in the default version of this file. They are straightforward, especially with respect to Figure 12. I list the defaults here, which you can change as the Linux administrative root user.

```
PASS_MAX_DAYS    99999
PASS_MIN_DAYS    0
PASS_MIN_LEN     5
PASS_WARN_AGE    7
```

- **PASS_MAX_DAYS** is the maximum number of days that new users are allowed to keep the same password.
- **PASS_MIN_DAYS** is the minimum number of days that a user has to keep the same password.
- **PASS_MIN_LEN** is the minimum length of a password, in alphanumeric characters.
- **PASS_WARN_AGE** is the number of days that a user gets a warning before his password is set to expire.

Synchronizing user databases

Linux is reliable. I stand by my belief that Linux is so reliable, that backup Domain controllers are rarely required. But there are things beyond the control of the Linux operating system that may require you to set up a different computer as the Domain controller for your network. For example, a power supply can fail, a hard drive can get bad sectors, or a network card can degrade. In any of these cases, you may be forced to bring up another computer as the PDC on your network.



If you need a backup computer for your Domain controller, you can set up Linux PDCs in a cluster. I describe clustering briefly in Chapter 9, "System Backup."

I'm assuming that you'll want to transfer just the files you need from a backup of your Linux PDC. And I'm assuming that you make regular backups. To set up another Linux computer as a Backup Domain Controller (BDC), you'll need to do the following:

- Copy the machine identifier from the PDC. Run the **smbpasswd -S** command, which takes the machine identifier from the PDC and stores it in the local /etc/samba/secrets.tdb file. (In Samba 3.0, the corresponding command is **net rpc getsid**.)
- Copy the Linux user database files from the PDC. This includes /etc/passwd, /etc/group, /etc/shadow, and /etc/gshadow.
- Copy the Samba user database files from the PDC. This includes /etc/samba/smbusers and /etc/samba/smbpasswd.
- Copy the Samba configuration file, /etc/samba/smb.conf. If you're going to use the backup computer as the WINS server, you may need to change the **wins server** variable in this file.
- Make sure the shares that you configured on the PDC are also available on the backup computer. If you have [netlogon] scripts or profiles on the PDC, you'll need to copy those files to the backup computer as well.

In the next chapter, I'll show you how to back up your PDC. You can restore these files from that backup. Alternatively, you can set up replication of these files in two ways. First, you can set up the prospective BDC as a Network Information Service (NIS) slave server, or as a Lightweight Directory Assistance Protocol (LDAP) server. Details of this process are beyond

the scope of this book. For more information, see the NIS or the LDAP HOWTOs available at **www.tldp.org/HOWTO/HOWTO-INDEX/howtos.html**.

Conclusion

While it's not possible to become a seasoned Linux administrator from a book of this size, I can provide some basic survival skills. To get around a Linux system, you do need to know how to use some basic navigational and file-management commands. If you ever have to rescue your system from a failure, you won't have access to a GUI and will need to know how to use the vi text editor.

Once you've acquired basic command-line skills, you don't need to stay in the office until the middle of the night to back up your computers. You can set up scripts to perform these tasks automatically.

If you ever have a problem booting Linux, you need to know something about the critical startup files: /etc/grub.conf, /etc/inittab, and /etc/fstab. Errors in these files can make Linux unbootable.

If you have an unbootable system, you can use the rescue floppy that you created during the installation process, or you can use the **linux rescue** mode available through the first Red Hat Linux installation CD.

Finally, you can do a bit more to manage users. You can set password policies with the Red Hat User Manager. And if you ever have a problem with your Linux PDC, now you know what you would need to copy to a second computer that you might want to configure as a backup PDC.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Chapter 9
System Backup

There are a lot of threats to your data. Viruses and attacks by malicious users can overwrite data. Power surges can erase data. Natural and man-made disasters can destroy the computers that store your data. Hard drives are mechanical devices that eventually fail, which also can affect your data. Users can accidentally erase important files such as airplane schematics and financial statements. For these reasons, users count on you as a system administrator to back up their data.

What you do to back up your computers depends on the value of your data and how far you're willing to go. You can hold users responsible for their own backups. Alternatively, you can create and store backups on floppy disks, CDs, tape drives, and so on. You can back up part or all of the data, even in real time.

You can configure a Redundant Array of Independent or Inexpensive Disks (RAID) to help you protect your data in real time. You can store important data on portable, writable media such as floppy disks and writable CDs. You can create and store backup data on remote computers on your network. You can even store the computers or portable media in remote physical locations.

In this chapter, I cover the basics of how RAID works, and how you can set it up on Red Hat Linux. I also describe the basic options for larger-scale hardware backups. I then show you the Linux commands that you can use to archive data, as well as the basic Microsoft Windows tools that you can use to back up data from a Linux hard disk. Finally, I show you how to put a Linux backup on a schedule using the **cron** daemon described in Chapter 8, "Administration and Management."

The commands described in this chapter are administrative; unless otherwise noted, I'm assuming that you've logged in to your Linux computer as the administrative root user.

Backup strategies and types

What you do to back up the computers on your network depends on the time and money you have available. If you're administering a time- and mission-critical network, you may want to set up a very high-speed connection to a remote site to replicate your data in real time. If your data is not as valuable, a weekly or even monthly backup may be sufficient. In addition, I'll show you two RAID arrays that can preserve your data on the local computer in real time.

In this chapter, I assume that you want to back up the contents of the /home directory. While it can help to back up the contents of other directories, /home by and large contains the data from your users.

In this basic scenario, if you lose a file server and need to restore it from scratch, you would take the following basic steps:

1. Fix the hardware. This may mean replacing the hard drive or configuring a new computer.
2. Reinstall the operating system on the computer. This basic process is covered in Chapters 1 and 2.

3. Reinstall any updates to the operating system. If it's Red Hat Linux, you'll need to use the Red Hat Update Agent described in Chapter 2, "Installing Linux as a File Server."
4. Reinstall any applications that you've added to the operating system—over and beyond any applications that you may have included when you installed Red Hat Linux (such as the OpenOffice.org suite).
5. Restore the data that you've backed up, presumably from the /home directory.

Backup strategies

The most straightforward backup involves copying all data on your hard drives. But hard drives now contain gigabytes and even terabytes of data. If you want to back up every byte of data on a larger hard drive, that can take hours. If you want to back up your hard drive on a remote computer, that can create the kind of traffic that could effectively shut down the network for your users.

Backups by system administrators are generally driven by four factors:

- Timing. You want to schedule backups during the hours when your users don't need your file server or network.
- Need. What you back up depends on the importance and timeliness of the data.
- Cost. Backing up everything on your hard drives will drive up the cost of backup media, as well as the demands on your network.
- Size. Larger backups are more difficult to manage. Depending on the speed of your network and the size of your hard drives, it may not be possible to completely back up your computer overnight, or even over a weekend.

Backup types

Most administrators use a combination of backup strategies. The files associated with the Linux operating system don't change unless you've used the Red Hat Update Agent, or have otherwise downloaded and installed new RPM packages. Therefore, you may not need to back up all of the files on your computer on a daily basis.

When setting up a backup, many administrators will back up everything on a hard drive weekly or monthly. If a hard drive is too large, they may limit the backup to all user files, which on a Linux computer can be found under the /home directory. They will then back up newly created files more often. To support these needs, there are three major types of backups: full, incremental, and differential.

Full backup

A full backup of a computer is not quite as straightforward as it sounds. A full backup does not have to include every file on a computer. The choice of full backup options is up to the administrator. For example, you can create a full backup of only a single hard disk on a computer. You can also create a full backup of a single directory on a Linux computer, such as /home.

Incremental backup

An incremental backup includes all of the files and directories that have been created or changed since the last full backup. It does not matter if you have other incremental or differential backups. As time goes on and more files are created or changed, incremental backups get larger. If you want to restore a system, you'll need to restore the files from the full backup and the latest incremental backup.

Differential backup

A differential backup includes all of the files and directories that have been created or changed since the last full, incremental, or differential backup. If you run differential backups on a daily basis, they'll typically remain about the same size. Unfortunately, restoring a system from a differential backup can get complex. If you want to restore a system with a differential backup, you'll need to restore the files from the latest full, incremental, and all subsequent differential backups.

RAID

Backing up and restoring data takes time. If you have a data disaster, your users may have to wait until you can restore their data from backups. There are alternatives. Several different types of RAID arrays can save data in real time. Your users can still access their data even if a single hard drive fails.

You can set up both hardware and software RAID arrays. A hardware RAID array consists of different physical hard drives. For that reason, the failure of a single hard drive does not cause you to lose all the data in that array. A software RAID array consists of different partitions. While you can configure a software RAID array with partitions on the same physical hard drive, I don't recommend it. If you do, any problem with that physical hard drive can lead to the loss of all your data on that array.



RAID is commonly used as an acronym for two different terms: a Redundant Array of Independent Disks, or a Redundant Array of Inexpensive Disks. I prefer the first acronym. Fault tolerance does not work unless the disks are physically independent.

Red Hat Linux supports three basic types of software RAID arrays, known as RAID 0, RAID 1, and RAID 5. An array is a group of hard drives or partitions configured together. A RAID 0 array is used to combine two or more hard drives or partitions in a single volume. While it is faster, it does not help to protect data. A RAID 1 array is used to configure two hard drives or partitions with identical data. A RAID 5 array is used to configure three or more hard drives or partitions, in a way that also protects your data.

RAID 0

A RAID 0 array is a group of two or more partitions or drives, grouped together. This grouping is known as a *volume*. If you set up a RAID 0 array on a Linux computer, you can mount a single directory such as /home on that volume.

If you set up a RAID 0 array on two different physical hard drives, your computer can use the buffers on each drive. That can speed reading and writing to that volume.

The drawback to RAID 0 is that it does not back up your data. If any disk or partition on a RAID 0 array fails, you lose all of the data that you've stored on that volume. RAID 0 is also known as "disk striping without parity."

RAID 1

A RAID 1 array is a group of two partitions or drives, paired together. Identical data is stored on both partitions or drives in this array. If one partition or drive is damaged, you can still access all of your data on the remaining partition or drive. For example, if you set up the /home directory on a RAID 1 array, you'll have identical copies of that directory on both drives or partitions.

Unfortunately, a RAID 1 array can be slower, because your data has to be written to both disks or partitions. And RAID 1 arrays can be expensive, because you'll need twice as much storage space for your data. RAID 1 is also known as "disk mirroring."

If you set up a spare disk or partition in a RAID 1 array and one component of the array is lost, Linux immediately can begin to rewrite the lost data to the spare disk.

RAID 5

A RAID 5 array is a group of three or more partitions or drives. Parity information is stored on each partition or drive in the array. If there's a failure in one component in the array, Linux can use the parity information to re-create the missing data. A RAID 5 array is also known as "disk striping with parity."

A RAID 5 array involves compromises. While it requires three or more partitions or drives, it sacrifices the space of only one partition or drive for parity information. If you include a spare partition or drive in a RAID 5 array, and one component of the array is lost, Linux can immediately begin to rewrite the lost data to the spare.

Configuring RAID on Red Hat Linux

You can set up a RAID array during the installation process as described in Chapter 1, "Basic Linux Installation." With the Red Hat Linux installer, you can set up a software RAID array on different partitions. **Figure 1** illustrates a software RAID array with partitions on four different physical hard drives.

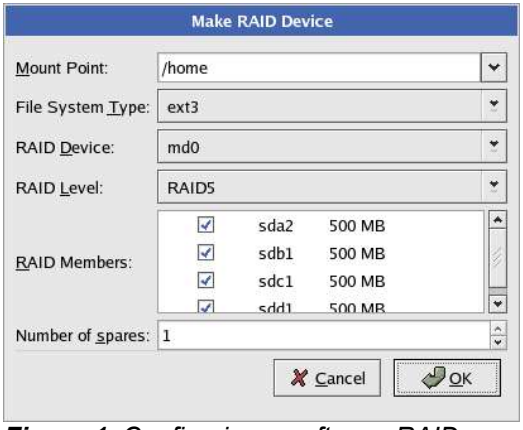


***Figure 1**. Configuring a software RAID array while installing Red Hat Linux.*

This particular configuration illustrates a RAID 5 array. I've included four partitions. The first three partitions (**sda2**, **sdb1**, and **sdc1**) are configured as part of the array. The last partition (**sdd1**) is reserved as a spare. Because all four partitions are located on different physical hard drives, the failure of any one physical hard drive does not destroy the data that you store in the array.

If you want to create or reconfigure a RAID array after installing Red Hat Linux, you'll need skills that are beyond the scope of this book. Because it is an important skill for a system administrator, I'll just outline the required steps here. A full description of the required steps is available in my book *Mastering Red Hat Linux 9*, published by Sybex.

1. Install one or more new physical hard drives.
2. After booting Linux, configure the new physical hard drives in a partition. Make sure that the size of the partition is about equal to the other partitions in the array. You can use either fdisk or parted for this purpose. Both utilities are located in the /sbin directory. I recommend fdisk as being more reliable. In fdisk, make sure to set the partition type to **fd**, which corresponds to a "Linux raid auto" type.



If you use fdisk or parted, be very careful. A small mistake with either utility can easily destroy all of the data on any partition or hard drive on your computer.

3. Format the partitions that you've just created. For example, if you've created the /dev/sde1 and /dev/sdf1 partitions to expand the size of the array, you'll need to run the following **mkfs** commands to format those partitions:

```
# mkfs -j /dev/sde1
# mkfs -j /dev/sdf1
```

4. Configure the partitions in a RAID array. If you've already installed a software RAID array, all you need to do is modify the /etc/raidtab configuration file. Red Hat Linux 9 includes a sample RAID configuration file in raidtab.sample, located in the /usr/share/doc/raidtools-1.00.3 directory. You can edit this sample file and save it as /etc/raidtab.
5. Create the RAID device. Typical RAID devices are based on file names such as /dev/md0 and /dev/md1. For example, if you're creating the third RAID device on your system, you'd run the following command:

```
# mkraid -R /dev/md2
```

6. Format the RAID device. This part is fairly straightforward, because it's analogous to the previous **mkfs** commands used to format individual partitions:

```
# mkfs -j /dev/md2
```

7. Now that you've created and formatted a RAID array, you're ready to mount the directory of your choice on that array. For example, if you've added partitions to increase the size of a RAID array for the /home directory, you could run the following commands. They copy the contents of the /home directory to the /tmp directory, after which you could mount the new RAID array.

```
# cp -ar /home /tmp
# mount /dev/md2 /home
```

8. Now you can copy the files back from /tmp to the original location.

```
# cp -ar /tmp/home /
```

9. Finally, you can set up the new /home directory in your /etc/fstab configuration file. Details of this process are also beyond the scope of this book. The next time you boot into Linux, it's necessary to make sure that Linux boots your /home directory on the appropriate RAID array.

Backup options

This section provides a high-level overview of the hardware backup options available for Linux and Windows. Much of the hardware is expensive, and the commands depend on what you might buy. A detailed discussion of these options could easily fill a book three times this size.

Media options

Choosing the media that you might use to back up your data depends on the amount of data that you want to keep safe. Individual users who work with text files may need only the space associated with a 1.44MB floppy disk. The same may hold true if you're backing up critical configuration files from the /etc directory, such as inittab, fstab, grub.conf, and the files in the /etc/samba subdirectory.



The /etc/grub.conf file is actually linked to the /boot/grub/grub.conf file. You can open and edit either file name; the result is saved in /boot/grub/grub.conf.

A number of other types of media are available that you can use with a Linux computer. The variations are endless; I cite only the "typical" size for each media type:

- Zip drives (from Iomega) are normally 100MB and are suitable for backing up other critical files such as the Linux kernel from the /boot directory. If you remember from Chapter 1, the standard Red Hat Linux installation allocates 100MB to the /boot directory.
- Bernoulli drives normally include 230MB of space, which might be enough to back up user files from the /home directory on some smaller networks.

- Writable CD media can store around 650MB of data, which can contain all but the data files for many dedicated Red Hat Linux servers where the GUI is not installed.
- Writable DVD media vary in size. They can contain 4.7GB to 17GB of data, which can be used to back up many Linux servers, including data files.
- Tape drives are available in a variety of sizes. As of this writing, I've seen single tape drives that can contain up to 300GB of data.

The media that you select depends on the money you have available, as well as your backup hardware. You also need to consider your specific environment. For example, if the only place where you can store your backups includes a number of magnetic fields (such as a factory or a machine shop), you may want to stick with CDs or DVDs.

System backups

There are a number of ways to back up larger file servers. The most direct way is to use third-party hardware to back up all the data on the server to some high-capacity media. You may also want to back up the hardware associated with your file server. This is possible courtesy of two different open-source projects: the High Availability Linux project and the Linux Virtual Server project.

Third-party hardware

If the hardware on your system, such as a CD or a DVD writer, is not enough, there are a number of third-party options available. For example, some systems can combine dozens of writable CDs, DVDs, and tape drives to save the data from every computer on your network.

A wide variety of hardware options are available. I use the data from **www.storagesearch.com** in my research. This site is published by Applied Computer Science, Ltd. of the United Kingdom. In addition to those previously described, some of the backup hardware types include:

- FireWire: One common option is to back up personal computers onto a portable high-capacity hard drive. This type of drive uses connections that correspond to the IEEE1394 standard, also known as iLink. This is essentially a high-speed SCSI connection, which I use to back up data at burst speeds of nearly 400 Mbps. Other types of FireWire drives can write data to other media such as writable DVDs. (Linux support for FireWire as well as the alternative USB 2.0 devices is experimental as of this writing.)
- Removable hard drives: If you've configured a RAID 1 array of removable hard drives, you can remove one of the drives in the array and store that drive in some secure remote location. If you have a spare hard drive in that array, a RAID 1 system automatically writes your data to that drive. You can then set up a new hard drive as a spare. Larger groups of removable hard drives are often configured as rack-mount storage.
- Jukeboxes: These are "storage cabinets" that allow you to combine the data capacity of a group of media types, such as writable tape, DVDs, and CDs.

- Storage area networks (SAN): A high-speed network of storage devices; many are in common use for higher-capacity Linux systems. While this isn't explicitly a backup option, it is a common way to organize a RAID system.

High Availability Linux project

The High Availability Linux project is one way to set up multiple computers as a *cluster*. The cluster of computers appears like a single server to the other computers on the network. They often share a common external storage medium, usually with a SCSI connection.

If one computer in the cluster fails, another computer takes over automatically. Client computers on the network don't know which computer in the cluster is working as the file server. Because the computers in the cluster share the same external storage, clients see no difference in the data.

While this does not back up your data, it does serve as a backup for your hardware. For example, if you set up a PDC in a cluster, the failure of one computer does not affect your network. For more information on the High Availability Linux project and its "heartbeat" software, navigate to **www.linux-ha.org**.

Linux Virtual Server project

The Linux Virtual Server project is another way to set up multiple computers as a cluster. This cluster of computers also appears like a single server to the other computers on the network. It also can be configured with a common external storage medium. One additional feature supports load balancing. In other words, if a lot of clients are working through one server in the cluster, load balancing sends additional clients to less busy servers in the cluster.

As with the High Availability Linux project's heartbeat software, if one computer fails, other computers take over automatically. The cluster of computers appears as one file server to the other computers on the network. For more information, see **www.linuxvirtualserver.org**.

Red Hat Enterprise Linux (RHEL) Advanced Server supports its own version of the Virtual Server project. Red Hat developed its Enterprise Linux distributions from older versions of Red Hat Linux for higher capacity systems. As of this writing, the latest version of RHEL Advanced Server is 2.1, which is based on Red Hat Linux 7.2.



RHEL 3 should be available by the time this book is released. RHEL 3 is based on the primary operating system used in this book, Red Hat Linux 9.

Backup and Restore commands

As of this writing, Red Hat Linux does not include any GUI tools for backing up or restoring data. However, the Nautilus file browser does support writing to a writable CD. In general, to perform backups, you'll need to work from the command-line interface. Text commands support scheduled backups through the **cron** daemon. Fortunately, the commands are not difficult. Once you've configured connections to remote computers, you can back up any directories shared from those computers.

Let's start by looking at some basic backup commands. You can create archives with the **tar** or **cpio** commands. You can **dump** and **restore** data to and from a tape drive. And you can record data to writable CDs and DVDs by using the **cdrecord** and **dvdrecord** commands.

In most cases, Linux backups involve a two-step process. First you need to create a file, such as an archive of the files or directory that you want to save. Then you can save the archive to media such as a tape drive or a writable CD.



No matter what method or media you use for backup, make sure you can restore it. The small amount of time you spend testing your backup media can save you a lot of frustration down the road.

tar archives

The **tar** command archives and records a group of files, usually the contents of a directory. With the right switches, you can collect a group of files into single compressed archive. In that way, the **tar** command is analogous to the WinZip utilities associated with Microsoft Windows.

Users can run the **tar** command to archive the files in their own directories. For example, if user **waymon** wanted to back up the files in his home directory to waymon.tar.gz, he could use the following command:

```
$ tar cvzf waymon.tar.gz /home/waymon
```

You may notice something a little strange with the **tar** command; it does not require a dash in front of the switches. There are several Linux commands that work this way; the **tar** command is the only one of this type that I cover in this book.

There are four switches associated with this particular command. It creates (**c**) the backup in the noted archive file (**f**), waymon.tar.gz. It runs the command in a verbose (**v**) way, which lists the files that are being collected in the archive. It then compresses the result (**z**), which reduces the space taken by the archive.

You may notice that **tar** archives are associated with two consecutive file extensions. The ".tar" indicates that it was created by the **tar** command. The ".gz" is associated with a compressed file.

Finally, it's important to cite the absolute path to the directory that you want to archive. Absolute paths start with the first forward slash, /. Linux reads this as starting from the top-level root directory (/). You can restore this archive from any location in the Linux directory tree; an archive created from an absolute path is automatically restored to the same location in the Linux directory filesystem hierarchy.

Naturally, you can reverse the process. The following command restores from the compressed waymon.tar.gz archive:

```
$ tar tkvzf waymon.tar.gz
```

This command lists (**t**) the files in the archive. It does not overwrite (**k**) any existing files. It works in verbose (**v**) mode so you can watch as **tar** restores from your archive. It works from a compressed (**z**) archive, from the (**f**) file cited, waymon.tar.gz.

The **tar** command is substantially more complex; for more information, open the **tar** manual from the command-line interface with the **man tar** command.

cpio archives

The **cpio** command can help you archive a group of files. The group can be in a single directory, or it can include all files with a single pattern of alphanumeric characters. The **cpio** command is almost literal, because it copies (**cp**) from input to output.

As I described in Chapter 8, the **find** command lists the files on your system, based on a search term. For example, the following command searches for all of the files on your computer with a JPG extension.

```
# find / -name *.jpg
```

You can now use the **cpio** command to collect these files. The following command combines **find** and **cpio**. The output from the **find** command is "piped" as input to the **cpio** command. The pipe is the vertical line next to or just under the backspace key on a standard U.S. keyboard:

```
# find / -name *.jpg | cpio
```



Linux's bash shell includes a number of characters that you can use to connect the input and output of different commands. The use of the following characters requires an understanding of standard input, standard output, and standard error: **>**, **>>**, **|**, **2>**, **<**. *This topic is beyond the scope of this book. You can find out more in my other book,* Mastering Red Hat Linux 9.

But this command is not complete. You have input from the **find** command, but you need a place for the output. The following command sends that output (**-o**) to the jpegs.cpio archive:

```
# find / -name *.jpg | cpio -o > jpegs.cpio
```

Alternatively, if you have a tape drive connected to your computer, you can send this output directly to that tape drive. Normally, the first tape drive on a Linux computer is associated with the /dev/st0 device file. Thus, you can archive all JPG files on your computer with the following command:

```
# find / -name *.jpg | cpio -o > /dev/st0
```

You could archive the files from your /etc/samba directory to a floppy drive. The first floppy drive on a Linux computer is normally associated with the /dev/fd0 device file. Thus, you can archive the files in your /etc/samba directory with the following command:

```
# find /etc/samba | cpio -o > /dev/fd0
```

The **cpio** command is substantially more complex; for more information, open the **cpio** manual from the command-line interface with the **man cpio** command.

Full, incremental, and differential backups

If you want to use the incremental and differential backup schemes described earlier in this chapter, use the **dump** and **restore** commands. If the amount of data is larger than the backup media, such as a tape or floppy drive, these commands support the use of multiple tapes or drives. As of this writing, Red Hat does not support the use of these commands with writable CDs or DVDs.

In short, the **dump** command archives data, and the **restore** command copies the data back from the archive. This section assumes you've installed a tape drive on your Linux computer, and it's installed on the /dev/nst0 device. You can check this on your own computer in your /etc/fstab configuration file.

dump archives

You can archive the directory of your choice by using the **dump** command. You can set up full, incremental, or differential backups with this command. Normally, the first step in this sequence is a full backup. For example, if you want to set up a full backup of the /home directory, you would run the following command:

```
# dump 0f /dev/nst0 /home
```

The 0f switch indicates the backup level. A "0" in this switch always results in a full backup. This number can vary between 0 and 9. Whether the **dump** command creates an incremental or a differential backup depends on the number associated with the previous **dump** command.

Remember, incremental backups save all files created or changed since the last full backup. Here's an example: If you want to create a series of five incremental backups, perhaps for each workday, you could run the numbers in the **dump** command backwards:

```
# dump 9f /dev/nst0 /home
# dump 8f /dev/nst0 /home
# dump 7f /dev/nst0 /home
# dump 6f /dev/nst0 /home
# dump 5f /dev/nst0 /home
```

The number that you start with does not matter. If the numbers go backward from 9, the next **dump** command creates an incremental backup.

In contrast, differential backups save all files created since the last backup of any kind. If you want to create a series of five differential backups, one for each workday, you would run the numbers forward:

```
# dump 1f /dev/nst0 /home
# dump 2f /dev/nst0 /home
# dump 3f /dev/nst0 /home
# dump 4f /dev/nst0 /home
# dump 5f /dev/nst0 /home
```

The numbers don't have to be in sequence; all that matters is that the number associated with the **dump** command is higher than the one you used with the previous **dump** command. As an example of how this works, let's see what happens if you created a full backup of the /home/waymon directory on two floppy disks. You'd start with the following command:

```
# dump 0f /dev/fd0 /home/waymon
```

You'll see a long series of messages related to the backup. If the files in /home/waymon are too large for a single floppy drive, **dump** returns the following messages:

```
DUMP: Change Volumes: Mount volume #2
DUMP: Is the new volume mounted and ready to go?: ("yes" or "no")
```

When you see the message, insert a new floppy and type **yes** at the prompt to continue the full backup process. There are a substantial number of options for the **dump** command, related to everything from compression to date labels. For more information, you can read the **dump** manual by typing in **man dump** at the command-line interface.

restore archives

One of the drawbacks of the **dump** command is that it assumes that you're starting from the top of the directory tree, the root directory (/). Thus, when you use the **restore** command, you need to make sure that you're in the top-level root directory (/) with the following command:

```
# cd /
```

Now you can restore from the device with the backup. I can restore from the backup that I created of the /home/waymon directory in the previous section. First I insert the first floppy disk of the backup, and then I run the following command:

```
# restore -rf /dev/fd0
```

This command restores (**-r**) the filesystem from the noted location (**-f**), in this case, the first floppy drive (/dev/fd0). If you haven't mounted /home/waymon on its own partition, you'll see a series of warning messages about other directories "not found on tape." If /home is mounted on its own partition, the **restore** command assumes that you're trying to restore every subdirectory under /home. You can safely ignore these messages.

Once the **restore** command finishes with this drive, it prompts for additional drives. For this particular backup, it gives me the following messages:

```
Mount volume 2
Enter "none" if there are no more volumes
otherwise enter volume name (default: /dev/fd0)
```

In this case, all I need to do is insert the second floppy of the backup into the drive and press Enter. If the rest of the backup is on another device such as /dev/nst0, I would enter that device name when prompted by the "otherwise enter volume name..." message.

Recording to CDs and DVDs

CD and DVD writers are fairly common, and often come installed with new computers. They can handle fairly large amounts of data. If you want to back up data to a writable CD or DVD, you'll need to take three basic steps:

1. Make sure Linux recognizes your hardware as a writable CD or DVD.
2. Archive the files that you want to save into a package that's suitable for writing to a CD or DVD.
3. Write the archive to the writable CD or DVD.

While most CDs are connected as IDE drives, Linux can only write data to a SCSI CD drive. In most cases, this is not a problem. When Red Hat Linux detects a writable IDE CD drive on your system, it sets up SCSI emulation. In some cases, Linux supports it with the following message to the GRUB boot loader, which I described in Chapter 8:

```
hdc=ide-scsi
```

If Linux can write to a CD, you should be able to see Linux detect the drive with the help of the **cdrecord --scanbus** or **dvdrecord --scanbus** commands:

```
# cdrecord --scanbus
Cdrecord 2.0 (i686-pc-linux-gnu) Copyright (C) 1995-2002 Jörg Schilling
Linux sg driver version: 3.1.24
Using libscg version 'schily-0.7'
cdrecord: Warning: using inofficial libscg transport code version (schily - Red
Hat-scsi-linux-sg.c-1.75-RH '@(#)scsi-linux-sg.c        1.75 02/10/21 Copyright
1997 J. Schilling').
scsibus0:
  0,0,0   0) 'LG      ' 'CD-RW CED-8083B ' '1.09' Removable CD-ROM
  0,1,0   1) *
  0,2,0   2) *
  0,3,0   3) *
  0,4,0   4) *
  0,5,0   5) *
  0,6,0   6) *
  0,7,0   7) *
```

Once you've confirmed a working writable CD or DVD drive, you can proceed to the next step, archiving the files that you want to save to a single file in ISO format. For example, if I wanted to save my /home directories in an ISO file, I'd run the following command:

```
# mkisofs -J -r -T -o /tmp/homecd.iso /home
```

This command is also a little strange. Unlike other commands, **mkisofs** requires that you set the switches with separate dashes. It sets up the Joliet (**-J**) filesystem, which makes the CD readable by a Microsoft operating system. It incorporates Rock Ridge (**-r**) extensions, which support the use of Linux files. It preserves the use of long file names (**-T**), and sends the output (**-o**) to the ISO file shown in the command. Pay attention to the messages at the end of the command, to make sure that the data you've archived does not exceed the capacity of the writable CD or DVD.

Because you might use this command to archive several GB of files, you may want to check the result before writing the ISO file to a CD or DVD. You can mount this file as if it were a CD or DVD with the following command:

```
# mount -t iso9660 -o loop /tmp/homecd.iso /mnt/cdrom
```

This particular command mounts the ISO file to a particular filesystem (**-t**), ISO 9660. This is the standard filesystem associated with CD and DVD data. With loop options (**-o loop**), it mounts a file instead of a device such as /dev/cdrom. Remember, when you're done checking the ISO file, you should unmount it with the following command:

```
# umount /mnt/cdrom
```

The spelling of this command is correct. It's **umount**, not *unmount*. Finally, you're ready to write the file to a writable CD or DVD. The commands are similar. For example, once you've inserted a blank writable CD, the following command writes the previously created ISO file to that CD.

```
# cdrecord -v speed=4 dev=0,0,0 /tmp/homecd.iso
```

The **cdrecord** command can be used with audio or data files. While the verbose (**-v**) switch is not required, it can help you diagnose any problems that arise. The **speed=4** switch sets the write speed to four times the CD standard, which is about 150 KBps for data. You should not set this above the rated speed of your CD writer. If you ever have problems reading from a CD that you create, try setting **speed** to a lower level. The **dev=0,0,0** switch corresponds to the output from the aforementioned **cdrecord --scanbus** command.

The **dvdrecord** command is similar. It works with ISO files as well. If the **dvdrecord --scanbus** command has detected a DVD writer on your computer, you can insert a blank DVD into the drive and run the following command:

```
# dvdrecord -v speed=1 -dao dev=0,1,0 /tmp/homecd.iso
```

This particular command runs in verbose (**-v**) mode, at the standard **speed=1** for writing to a DVD drive, which is about 1.3 MBps for data. You should not set this above the rated speed of your DVD writer. The **dev=0,1,0** switch corresponds to the output from the aforementioned **dvdrecord --scanbus** command.

A GUI CD recording utility

Alternatively, you can use the Red Hat Linux Nautilus browser to copy and back up files to a writable CD. Log in to the Red Hat Linux GUI as the root user. Insert a blank CD into a CD writer. Nautilus should come up automatically with the **burn:**/// label. You can then open another Nautilus window and drag the files or directory folders of your choice. **Figure 2** illustrates my backup of the /home directories to a blank CD.



You may have heard of the Linux CD-Roast utility. It has become something of a standard for recording data on CDs. Unfortunately, it is still beta software and is therefore something that I'm not allowed to cover in this book.

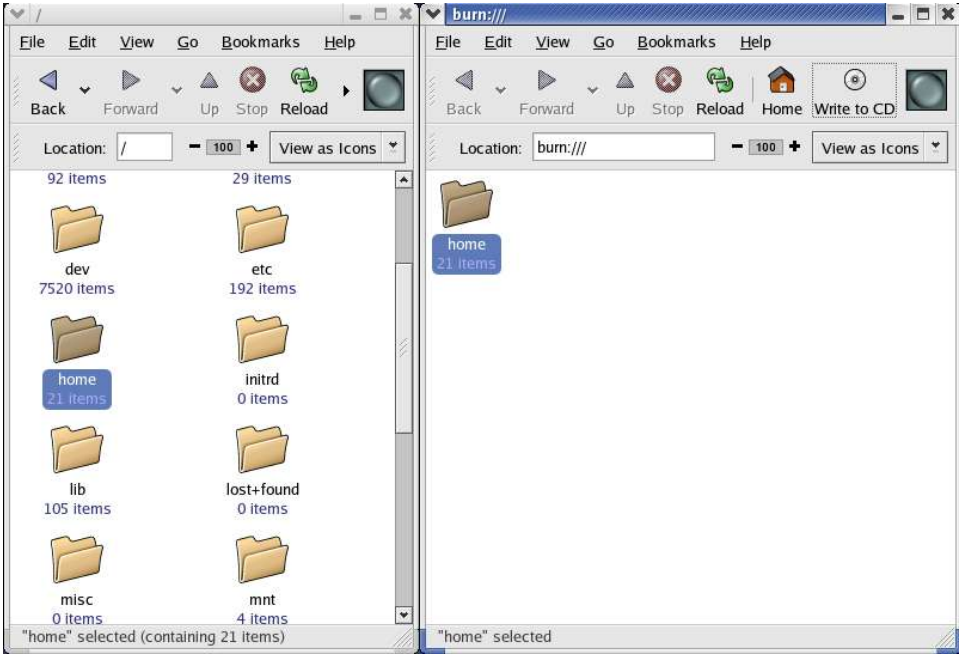


***Figure 2**. Dragging the /home directory to a writable CD.*

When I've finished with what I want to back up, I click the Write to CD button. This opens the window shown in **Figure 3**. If you have more than one writable CD on the local computer, you can select a different "Target to write to:". You can also vary the speed with which the data is written, change the label for the CD, and save the write list if you want to create more than one CD from this data.

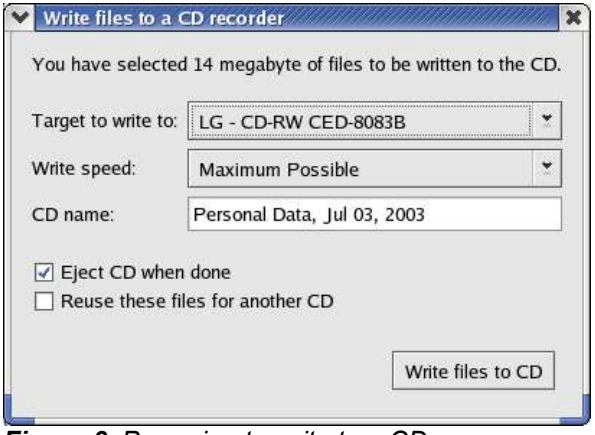


***Figure 3**. Preparing to write to a CD.*

Backing up over a network

One of the reasons for having a file server is so that you have a central database of files for easier backups. Users on your network can store all important data on the file server, and then as the administrator, all you have to do is back up user directories on that file server. I illustrated this process in the previous section of this chapter.

If the Linux partition with the /home directory is large enough, the users on your network can simply save their important files to their individual home directories. Your work to back up the /home directory then automatically backs up your users' files. As usual, there are four basic scenarios for backups on a mixed network with Linux and Windows computers:

- From a Windows client to a Windows file server.
- From a Windows client to a Linux file server.
- From a Linux client to a Windows file server.
- From a Linux client to a Linux file server.

The basic steps that you take to connect from a Linux client are the same, whether you're connecting to a Windows or a Linux file server. I've described some of the differences in Chapter 5, "Connecting Windows Workstations." For our purposes, some defaults such as home directories do change from server to server, depending on how the Windows and Linux file servers are configured.

As with the rest of the book, I'm assuming that you're using Samba on your Linux computers to connect to other Linux and Windows computers in a Microsoft Windows-style network.

If the file server resides on a Linux or Windows Domain Member Server, you can share and back up from the directories of your choice. If it's a Linux Domain Member Server, you can use the techniques that I describe in Chapters 3 and 4 to share its /home directory for read-only backups over the network.

Backing up from a Windows client to a Windows file server

Because this book is written for Microsoft Windows administrators, I'm assuming that you know how to configure a backup from a Windows client to a Windows file server. I'll just review the basic technique because you can use the same techniques to back up Linux clients.

Remember, the home directories and profiles are associated with the Microsoft Windows User Profile. In Windows NT 4 Server, the User Environment Profile is associated with the User Manager for Domains, as shown in **Figure 4**.

Naturally, you should configure user home directories and profiles in the same folders. This will ease your burden when you back up these files. Windows NT 4 Server includes its own backup utility, which is accessible from Start | Programs | Administrative Tools (Common) | Backup. (Windows 2000 Server includes a similar backup utility when you click Start | Programs | Accessories | System Tools | Backup.) Because this is a book on Linux, I don't address backups from a Windows client to a Windows server in detail.

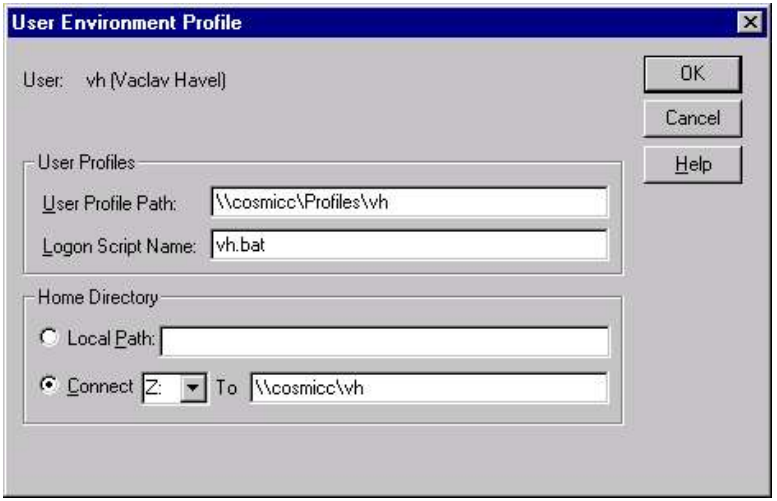


***Figure 4**. Defining user files and directories in Windows NT 4 Server.*



*Windows NT 4 Server's backup utility works only with tape drives. If you use other backup media, you'll need a third-party utility to support your backup. There is a seemingly endless series of third-party backup utilities listed in the Google Web directory at **directory.google.com/Top/Computers/Software/Backup**.*

Backing up from a Windows client to a Linux file server

In Chapters 3 and 4, I configured a number of commands in the main Samba configuration file, /etc/samba/smb.conf. These commands configure user home directories and Windows profiles within the /home directory. Therefore, users who log in to a Domain on a Linux PDC can save the files of their choice to their home directories.

Once you've logged in to a Linux PDC, you have access to your home directory on the PDC or any Linux member servers through the Windows Network Neighborhood. If you access it through Windows Explorer, it's easy to back up the files and directories of your choice to your home directory. **Figure 5** illustrates my connection to the /home/mj directory from my Windows XP Workstation client.

Alternatively, if you're willing to create an additional shared directory from your Microsoft Windows client, you can connect to it from the Linux file server by using the **smbmount** command described in Chapter 5, "Connecting Linux Workstations." Once you've mounted the shared directory from the Microsoft client, you can work with the shared Windows directory as if it were local to the Linux file server. You can then back up the files from that directory using the text commands described earlier in this chapter. If you're backing up to a writable CD, you can even use the GUI Nautilus tool shown in Figure 2.

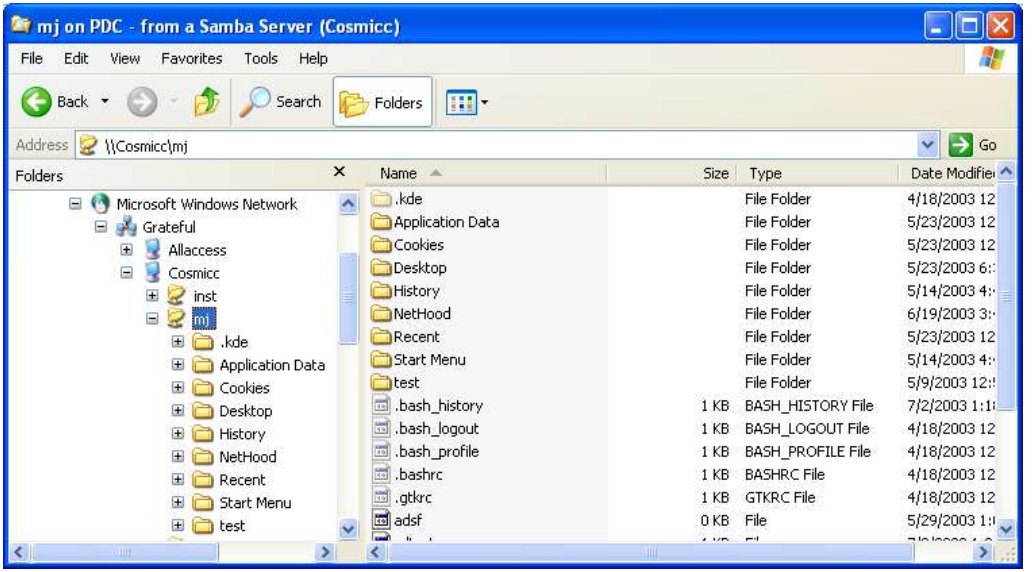


***Figure 5**. A connection to a user home directory on a PDC file server.*

Backing up from a Linux client

As described in Chapter 5, when you log in to a Linux client computer on a Domain, the user name and password are normally local to the client. In that case, when you connect to a Domain PDC or member server, you'll need to enter the Domain user name and password separately.

For example, in the Linux GUI, you can view the computers on your Domain through Nautilus. Click Main Menu | Network Servers. This opens a Nautilus window pointing to **smb:///** to view the Domain. Once you click on the Domain and then select the PDC or member server, you'll be prompted for a Domain user name and password in the Authentication Required dialog shown in **Figure 6**.



***Figure 6**. Connecting to a Domain server.*

Once you've connected to the file server, you'll have access to your home directory. As you can see in **Figure 7**, there are two Nautilus windows. The left-hand window includes the

files in the local /home/mj directory. The right-hand window includes the files in the **mj** share from the **cosmicc** computer. Based on the [homes] share in smb.conf described in Chapters 3 and 4, you should recognize this as the share from the /home/mj directory on the **cosmicc** computer.

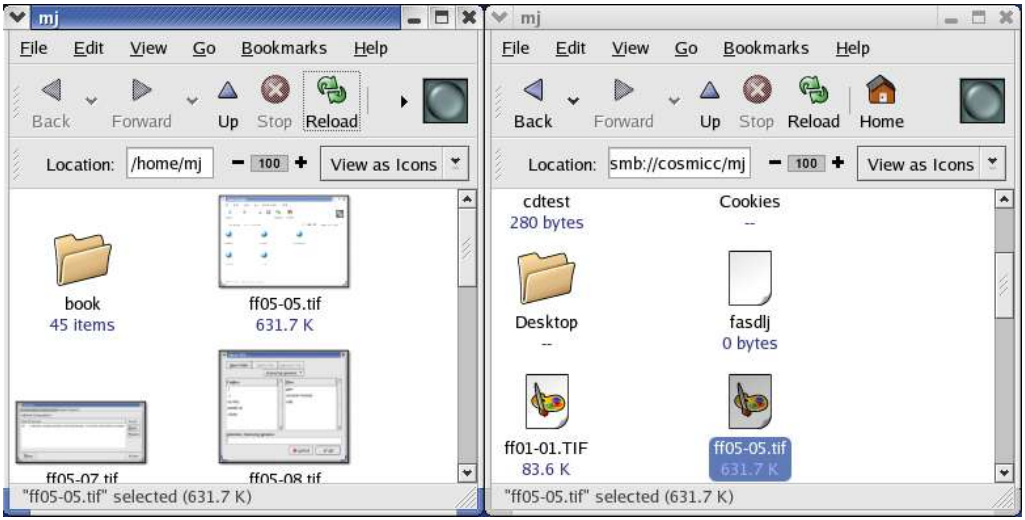


***Figure 7**. Side-by-side local and file server home directories for user **mj**.*

You can now click and drag the files and directories of your choice from the local /home/mj directory to the /home/mj directory on the **cosmicc** file server. For the example shown in Figure 7, you could click and drag the /home/mj/book directory to the **cosmicc** file server. When you do, Nautilus takes the time it needs to copy these files over your network.

Scheduled backups

In Chapter 8, I showed you how Linux's **cron** service runs scripts on a regular basis. Now I'll show you how to make it work for backups. As an administrator, you'll want to make sure to back up at least the files in the /home directory on a regular basis. Based on what you learned in Chapter 8, I can show you how to create a script you need.

Creating a file server script

Linux scripts are text files that you can create with GUI or command-line text editors. As described in Chapter 8, the slocate.cron script in the /etc/cron.daily directory starts with the following command, which sets this script to use the bash shell.

```
#!/bin/sh
```



A wide variety of command shells are available, but bash is the default shell for Red Hat Linux.

Let's say you're setting up a full backup of the /home directory to a tape drive, such as /dev/nst0. In that case, you'll want to add the following command to your script:

```
/sbin/dump 0f /dev/nst0 /home
```

Of course, this will work only if you actually have a tape drive connected to the /dev/nst0 device. You'll note that this is a bit different from the command described earlier in this chapter, which did not include **/sbin** before the **dump** command. That's because scripts require the full path to any commands that you use, unless you've set up a **PATH** command similar to the one I explained for the /etc/crontab script in Chapter 8, "Administration and Management."

The previous command is all you need in the script. Save it with an appropriately descriptive file name such as /etc/cron.weekly/fullbak. As described in Chapter 8, Red Hat Linux runs scripts in the /etc/cron.weekly directory every Sunday morning at 4:22.

But the script isn't ready until you make it executable. To do so, you'll need to use the **chmod** command described in Chapter 4, "Setting Up Your File Server's Users." The following command makes the script readable and executable by all users:

```
# chmod 755 /etc/cron.weekly/fullbak
```

You can create similar scripts for incremental or differential backups, based on your own desired schedule. The /etc/crontab script may not be flexible enough for your needs. As explained in Chapter 8, this script executes every script in the /etc/cron.weekly directory every Sunday.

For example, say you wanted to create an incremental backup every Wednesday night at 11:30. You'd need to set up a new directory, such as /etc/cron.wednesday. You could then add the following command to /etc/crontab:

```
30 23 * * 3 root run-parts /etc/cron.wednesday
```

Then you could create the script you need for incremental backups, and save it in the /etc/cron.wednesday directory.

Other commands

The aforementioned script is just a sample of what you can do. You're not limited to the **dump** command. Depending on the hardware you have available, you can configure the other commands in this chapter in scripts that back up your /home directory to appropriate media such as writable CDs or DVDs.

As of this writing, Linux does not support automation of GUI utilities. Therefore, you need to use text commands to back up the /home directory from a Linux file server.

Conclusion

File servers are a central location for user files on a network. Because they contain valuable information from your users, you'll want to back up your file severs on a regular basis. By definition, Linux computers save user files in the /home directory.

What you do to back up your file servers depends on a number of factors, including the amount of data, the frequency of required backups, and the media that you select. You can set up RAID arrays to keep data available on a real-time basis. You can back up all or part of your

data. You can set up full, incremental, and differential backups. There are a number of third-party hardware options that can support different levels of hardware RAID and large-scale backups.

You can even set up backups for your Linux file server hardware with clusters and virtual servers. This is supported through the High Availability Linux project and the Linux Virtual Server project.

Red Hat Linux supports three levels of software RAID: RAID 0, RAID 1, and RAID 5. You can set up a RAID array during the installation process, and can configure additional RAID partitions with fdisk or parted. You can set up the RAID array in larger or additional arrays in the /etc/raidtab configuration file.

To set up backups from Linux, you need to learn a number of different text commands. While you can copy files through the GUI to blank CDs, that does not help you with tape drives, DVDs, or other media. Because you can't automate Linux GUIs, you need to learn the basic text commands used for backups, including **dump**, **restore**, **tar**, and **cpio**. You may also find the commands that send data to writable CDs and DVDs valuable. These include **mkisofs**, **cdrecord**, and **dvdrecord**.

Your network users should save all their important data to your file servers. Then all you need to do is back up the appropriate home directories from these servers. These home directories on Windows servers are defined in user profiles. These home directories on Linux servers are located in the /home directory.

You don't want to back up a file server when it's busy. And you don't want to stay up in the middle of the night to back up files when the servers aren't busy. From a Linux server, you can automate backups with scripts. You can set up the scripts you need with the existing structure defined in /etc/crontab. You can create and store daily and weekly scripts in the /etc/cron.daily and /etc/cron.weekly directories. You can even create your own directories for any specialized scripts that you may need.

Updates and corrections to this chapter can be found on Hentzenwerke's Web site, **www.hentzenwerke.com**. Click "Catalog" and navigate to the page for this book.

Appendix A
Samba 3.0 Preview

Samba 3.0 was officially released in September of 2003. It is not included with Red Hat Linux 9. You can use it to more tightly integrate your Linux computers in a network with Microsoft Windows computers. The Samba designers have included a number of tools that allow Linux computers to work as Active Directory member servers with other Windows 2000/2003 Servers. You can download this software from www.samba.org and the Fedora Rawhide database currently at ftp.redhat.com.

In this book, I've covered using Red Hat Linux 9 with Samba 2.2.7 to serve as a File and Print server on a network with Microsoft Windows computers. This network is the standard Microsoft network, which uses Server Message Blocks (SMB) over the NetBIOS protocol.

You can use a Linux computer so equipped as a client on a peer-to-peer Workgroup or a Domain. You can configure that computer as a member server on a peer-to-peer Workgroup or a Domain. You can even configure that computer as a Primary Domain Controller (PDC) on a Domain. That computer can take the place of a Microsoft Windows NT 4 PDC. Unfortunately, it cannot take the place of a Domain controller in a network associated with the Active Directory features of Windows 2000/2003 Server.

Because Samba 3.0 was not included with Red Hat Linux 9, a full description of this software is beyond the scope of this book.

New features in Samba 3.0

There are a number of new features in Samba 3.0 that bring Linux into the standard network environment associated with Microsoft Windows 2000/2003 Server. Samba 3.0 has been under active development for years. As Microsoft has released new features in Windows 2000/2003, the Samba developers have worked on matching those features. These features include:

- Access Control Lists (ACL). Samba 3.0 supports the use of ACL, as long as you're working with a format such as the ext3 system associated with Linux.
- Active Directory support. Samba 3.0 allows a Linux computer to join an Active Directory realm as a member server. This is possible with the support that the Samba developers are including for LDAP and Kerberos 5.
- Authentication flexibility. Samba 3.0 allows a Linux administrator to manage Samba-enabled users, groups, and computers through the Samba configuration file, smb.conf.
- File name mangling support. File name mangling is the way that long file names are represented as MS-DOS files. For example, file name mangling in MS-DOS represents a long file name such as linuxtransferchapter1.doc as linuxt~1.doc.
- DOS-style **net** commands. Samba 3.0 includes **net** commands that can help you view and manage the users, groups, and computers on your network. While they are not identical to the **net** commands in MS-DOS, they are more capable.

- Microsoft Windows NT-style error codes. Samba 3.0 includes support for Windows NT error codes and messages such as "System error 53 has occurred" when a computer cannot be found.
- Additional printer details. Samba 3.0 supports sharing additional attributes of shared printers in the active directory.
- **Winbind** daemon flexibility. The Samba **winbind** daemon supports authentication through the Domain controller with the user name and password authentication database. Samba 3.0 supports an upgraded **winbind** with two threads. One thread keeps the authentication database up to date. The other thread answers requests from the cache. This dual-thread configuration promises to improve performance.
- Easier migration from a Windows NT 4 PDC. Samba 3.0 includes a migration script that eases the burden of converting from a Windows NT 4 PDC to one based on Samba on a Linux computer. This script will convert users and groups, as well as the machine Security Identifiers (SID) for the Domain.
- **Winbind** architecture mapping. Samba 3.0 upgrades the **winbind** daemon architecture to support mapping between a machine SID and user and group IDs. This will require LDAP support.
- Alternate user name/password database. For stand-alone servers, such as on a peer-to-peer Workgroup, you can set up a password database suitable for Microsoft Windows computers in the /etc/samba/passwd.tdb file.
- Trust relationships. Samba 3.0 supports the creation of trust relationships between Windows NT 4 Domains. If you have a large network, you probably have multiple Domains. One of the **net** commands will support the creation of trusts between Domains.

New variables in the Samba configuration file

As with the current version of Samba, the behavior of Samba 3.0 and Linux depends on the configuration as defined in the main Samba configuration file. In Red Hat Linux 9, that file is located in /etc/samba/smb.conf. Samba 3.0 includes a number of new defaults as well as newly configurable variables.

The following highlights will give you more of a feel of what you can do with Samba 3.0. The behavior of some of the variables that you've seen in this book changes for Samba 3.0.

- **encrypt passwords**: Passwords are now encrypted by default. If you want to accommodate those older Windows operating systems that can't handle encrypted passwords, you'll have to add the following line to smb.conf:

```
encrypt passwords = no
```

- **winbind uid**, **winbind gid**: These variables have been replaced by **idmap uid** and **idmap gid**. The new variables support mapping to NT user and group SIDs.

- **add user script**: This command is replaced by **add machine script**; the default version of this command no longer assigns the computer account to any specific group.

Naturally, there are a number of new variables that you might find useful when you incorporate Samba 3.0 into your network. Some of the more interesting variables related to user names and passwords include the following:

- **add group script**: Supports creation of Windows Domain groups on the local Linux computer. Closely related to **delete group script**.
- **add user to group script**: Supports adding a Samba-enabled user to a Windows Domain group on the local Linux computer. Closely related to **delete user from group script**.
- **auth methods**: If you have multiple user name/password authentication databases, this variable allows you to set the search order.
- **passdb backend**: Allows you to switch between password databases. The default uses the **smbpasswd** command. You can also set up this variable to look at the passwd.tdb database for a stand-alone server, or a database that conforms to LDAP.
- **realm**: Lets you specify the Kerberos 5 server to use for authentication with an associated LDAP database.
- **set primary group script**: Allows administrators to set a primary group for a new user.

You can also use several new Samba configuration variables to manage the files that are shared with clients on the network. Some of these variables include the following:

- **hide special files**: Linux includes a number of special files that you may not want to share, such as devices.
- **hide unwritable files**: You can minimize the frustration of some users by hiding read-only files from browse lists. Naturally, this is not a good idea if you've shared a directory that is supposed to be read-only.
- **map acl inherit**: Supports mapping from a Microsoft Windows ACL.

Conclusion

This book is focused on Samba 2.2.7, which is included with Red Hat Linux 9. It allows you to set up a Linux computer in one of three roles on a Microsoft Windows network: a client, a member server of a Domain or a peer-to-peer Workgroup, or a PDC that can substitute for a Windows NT 4 computer.

Two major factors might make Samba 3.0 valuable to you. It allows you to configure Linux as a server on a Microsoft Active Directory network, which makes it a suitable replacement for a Windows 2000/2003 server. It also allows you to automate upgrades from a Domain based on a Windows NT 4 PDC.

Appendix B Sample Samba Configuration Files

This appendix lists a couple of different versions of the main Samba configuration file, /etc/samba/smb.conf. The first file is used on a PDC, the second on a Domain member server. The comments are a mix of those normally included with the standard smb.conf file, and those based on variables that I've added in this book. In Red Hat Linux, the standard Samba configuration files are located in the /etc/samba directory.

Primary Domain Controller

The Primary Domain Controller (PDC) includes the database of user names and passwords for a Domain. With the following smb.conf configuration file, you can set up a Linux computer with Samba in place of a Windows NT 4-based PDC. The following file is based on the default version of the file that comes with the Red Hat Linux 9 Samba packages. I've deleted those commands that are not used; I've added some comments in caps. Near the end of the file, I've also added a couple of shares that I configure on my own computers. For a more detailed explanation, see Chapters 3 and 4.



The default Samba configuration file includes misspelled words and imperfect grammar. Don't let that bother you.

```
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options (perhaps too
# many!) most of which are not shown in this example
#
# TO ACCESS THE smb.conf MANUAL PAGE, RUN THE man smb.conf COMMAND.
#
# Any line which starts with a ; (semi-colon) or a # (hash)
# is a comment and is ignored. In this example we will use a #
# for comments and a ; for parts of the config file that you
# may wish to enable
#
# NOTE: Whenever you modify this file you should run the command "testparm"
# to check that you have not made any basic syntax errors.
#
#======================= Global Settings =====================================
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
   workgroup = GRATEFUL

# LOCAL COMPUTER NAME
   netbios name = cosmicc
```

```
# server string is the equivalent of the NT Description field
   server string = PDC - from a Samba Server

# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
   printcap name = /etc/printcap
   load printers = yes

# It should not be necessary to spell out the print system type unless
# yours is non-standard. Currently supported print systems include:
# bsd, sysv, plp, lprng, aix, hpux, qnx, cups
# CUPS IS THE DEFAULT FOR RED HAT LINUX 9
   printing = cups

# Uncomment this if you want a guest account, you must add this to /etc/passwd
# otherwise the user "nobody" is used
  guest account = pcguest

# THIS COMMAND ADDS COMPUTER ACCOUNTS FOR LINUX AND WINDOWS NT/2000/XP CLIENTS
# TO BECOME A PART OF THE DOMAIN; NOT REQUIRED FOR WINDOWS 9X/ME CLIENTS.
add user script = /usr/sbin/useradd -d /dev/null -g 100 -s /bin/false -M %u

# this tells Samba to use a separate log file for each machine
# that connects to this Samba server
# THIS CAN HELP IDENTIFY CONNECTION PROBLEMS FROM A SPECIFIC COMPUTER
   log file = /var/log/samba/%m.log

# Put a capping on the size of the log files (in Kb).
# A MAX OF 0 CORRESPONDS TO NO LIMIT
   max log size = 0

# Security mode. Most people will want user level security. See
# security_level.txt for details.
# THIS IS REQUIRED FOR A PDC; MEMBER SERVERS REQUIRE security = domain
   security = user

# You may wish to use password encryption. Please read
# ENCRYPTION.txt, Win95.txt and WinNT.txt in the Samba documentation.
# Do not enable this option unless you have read those documents
# ENCRYPTED PASSWORDS ARE THE DEFAULT FOR LINUX, WINDOWS 95 (WITH OSR2),
# WINDOWS 98/ME, WINDOWS 2000/XP/2003
   encrypt passwords = yes
   smb passwd file = /etc/samba/smbpasswd

# The following are needed to allow password changing from Windows to
# update the Linux system password also.
# NOTE: Use these with 'encrypt passwords' and 'smb passwd file' above.
# NOTE2: You do NOT need these to allow workstations to change only
#        the encrypted SMB passwords. They allow the Unix password
#        to be kept in sync with the SMB password.
   unix password sync = Yes
   passwd program = /usr/bin/passwd %u
   passwd chat = *New*password* %n\n *Retype*new*password* %n\n
*passwd:*all*authentication*tokens*updated*successfully*

# You can use PAM's password change control flag for Samba. If
# enabled, then PAM will be used for password changes when requested
# by an SMB client instead of the program listed in passwd program.
```

```
# It should be possible to enable this without changing your passwd
# chat parameter for most setups.
   pam password change = yes

# Unix users can map to different SMB User names
# WITHOUT THIS COMMAND, WINDOWS USERS WOULD REQUIRE LINUX USERNAMES
  username map = /etc/samba/smbusers

# This parameter will control whether or not Samba should obey PAM's
# account and session management directives. The default behavior is
# to use PAM for clear text authentication only and to ignore any
# account or session management. Note that Samba always ignores PAM
# for authentication in the case of encrypt passwords = yes
  obey pam restrictions = yes

# Most people will find that this option gives better performance.
# See speed.txt and the manual pages for details
   socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

# Browser Control Options:
# set local master to no if you don't want Samba to become a master
# browser on your network. Otherwise the normal election rules apply
# THIS SETS UP THE PDC AS THE MASTER BROWSER FOR THE LAN
local master = yes

# OS Level determines the precedence of this server in master browser
# elections. The default value should be reasonable
# THE DEFAULT IS 20; STANDARD FOR AN NT PDC = 32; MAX = 255
   os level = 63

# Domain Master specifies Samba to be the Domain Master Browser. This
# allows Samba to collate browse lists between subnets. Don't use this
# if you already have a Windows NT domain controller doing this job
   domain master = yes

# Preferred Master causes Samba to force a local browser election on startup
# and gives it a slightly higher chance of winning the election
   preferred master = yes

# Enable this if you want Samba to be a domain logon server for
# Windows95 workstations.
# THIS COMMAND APPLIES FOR ALL WINDOWS WORKSTATIONS
   domain logons = yes

# If you enable domain logons then you may want a per-machine or
# per user logon script.
# Run a specific logon batch file per workstation (machine).
;   logon script = %m.bat
# Run a specific logon batch file per username.
# THE LOCATION OF THESE SCRIPTS DEPEND ON THE [Netlogon] SHARE
   logon script = %U.bat

# Where to store roving profiles (only for Win95 and WinNT)
#        %L substitutes for this servers netbios name, %U is username
#        You must uncomment the [Profiles] share below
# WINDOWS 9X PROFILES ARE STORED IN USERS' HOME DIRECTORIES;
# WINDOWS NT/2000/XP PROFILES ARE STORED IN THE /home/Profiles/$USER DIRECTORY.
   logon path = \\%L\Profiles\%U
;  logon path = \\%L\Profiles\%U\%M
```

```
# Windows Internet Name Serving Support Section:
# WINS Support - Tells the NMBD component of Samba to enable it's WINS Server
# THIS IS ALL YOU NEED TO ACTIVATE A WINS SERVER WITH SAMBA
   wins support = yes

# DNS Proxy - tells Samba whether or not to try to resolve NetBIOS names
# via DNS nslookups. The built-in default for versions 1.9.17 is yes,
# this has been changed in version 1.9.18 to no.
   dns proxy = no

#============================ Share Definitions ==============================
# THIS IS THE STANDARD SHARE FOR USER'S HOME DIRECTORIES
[homes]
   comment = Home Directories
   browseable = no
   writable = yes
   valid users = %S
   create mode = 0664
   directory mode = 0775
# If you want users samba doesn't recognize to be mapped to a guest user
; map to guest = bad user

# THIS SETS UP LOGON DIRECTORIES FOR USER BATCH FILES IN
# /home/netlogon/username
 [netlogon]
   comment = Network Logon Service
   path = /home/netlogon
   guest ok = no
   writable = no
;   share modes = no


# THE FOLLOWING SHARE STORES WINDOWS NT/2000/XP PROFILES IN
# /home/profiles/username
# WINDOWS 9X/ME PROFILES ARE STORED IN USER HOME DIRECTORIES
[Profiles]
    comment = Roving profiles
    path = /home/profiles
    read only = no
    create mode = 0600
    directory mode = 0700
    browseable = no
    guest ok = no

# NOTE: If you have a BSD-style print system there is no need to
# specifically define each individual printer
# THIS SHARES CUPS PRINTERS AS LISTED IN /etc/printcap
[printers]
   comment = All Printers
   path = /var/spool/samba
   browseable = no
# Set public = yes to allow user 'guest account' to print
   guest ok = no
   writable = no
   printable = yes

# This one is useful for people to share files
# THIS IS A NORMAL SHARE FOR DOWNLOADS
```

```
[tmp]
   comment = Temporary file space
   path = /tmp
   read only = no
   public = yes

# THIS SHARE IS USEFUL IF YOU WANT TO DISABLE ENCRYPTION; REGISTRY SCRIPTS ARE
# STORED IN THE NOTED DIRECTORY.
[registry]
    comment = Disabing Encryption
    path = /usr/share/doc/samba-2.2.7a/docs/Registry
    writable = no

# IF YOU SET UP RED HAT LINUX INSTALLATION FILES IN /mnt/source, YOU CAN SHARE
# THE SOURCE WITH SAMBA
[inst]
  comment = Red Hat Linux 9 CD files
  path = /mnt/source
  browseable = yes
  guest ok = yes
```

Domain member server

A Domain member server relies on a PDC for the database of user names and passwords for a Domain. With the following smb.conf configuration file, you can set up a Linux computer with Samba in place of any Microsoft Windows computer that shares directories or printers on an NT-style Domain that does not include Active Directory. This is based on the default version of the file that comes with the Red Hat Samba packages. I've deleted those commands that are not used; I've added some comments in caps. I've also added a couple of shared directories to the end of the file. For a more detailed explanation, see Chapters 3 and 4.



The default Samba configuration file includes misspelled words and imperfect grammar. Don't let that bother you.

```
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options (perhaps too
# many!) most of which are not shown in this example
#
# TO ACCESS THE smb.conf MANUAL PAGE, RUN THE man smb.conf COMMAND.
#
# Any line which starts with a ; (semi-colon) or a # (hash)
# is a comment and is ignored. In this example we will use a #
# for commentry and a ; for parts of the config file that you
# may wish to enable
#
# NOTE: Whenever you modify this file you should run the command "testparm"
# to check that you have not made any basic syntactic errors.
#
```

```
#======================= Global Settings =====================================
[global]

# workgroup = NT-Domain-Name or Workgroup-Name
  workgroup = grateful

# server string is the equivalent of the NT Description field
  server string = experimental samba server

# NAME OF THIS COMPUTER
  netbios name = nopaws

# if you want to automatically load your printer list rather
# than setting them up individually then you'll need this
  printcap name = /etc/printcap
  load printers = yes

# It should not be necessary to spell out the print system type unless
# yours is non-standard. Currently supported print systems include:
# bsd, sysv, plp, lprng, aix, hpux, qnx, cups
# CUPS IS THE DEFAULT FOR RED HAT LINUX 9
  printing = cups

# this tells Samba to use a separate log file for each machine
# that connects
# YOU CAN CHECK CONNECTION LOGS BASED ON THE COMPUTER NAME IN THE
# /var/log/samba DIRECTORY
  log file = /var/log/samba/%m.log

# Put a capping on the size of the log files (in Kb).
# CORRESPONDS TO NO LIMIT
  max log size = 0

# Security mode. Most people will want user level security. See
# security_level.txt for details.
# THIS IS REQURIED FOR ANY SERVERS THAT ARE PART OF A DOMAIN -
# UNLESS IT'S THE PDC
  security = domain

# Use password server option only with security = server
# The argument list may include:
#   password server = My_PDC_Name [My_BDC_Name] [My_Next_BDC_Name]
# or to auto-locate the domain controller/s
#   password server = *
;   password server = <NT-Server-Name>
# SPECIFIES THE NETBIOS NAME OF THE PDC; password server = * SHOULD ALSO WORK
  password server = cosmicc

# THESE COMMANDS ASSIGN UIDs AND GIDs FROM THE USERNAME DATABASE FROM A PDC
# REQUIRES AN ACTIVE WINBIND DAEMON
  winbind uid = 5000-6000
  winbind gid = 5000-6000

# You may wish to use password encryption. Please read
# ENCRYPTION.txt, Win95.txt and WinNT.txt in the Samba documentation.
# Do not enable this option unless you have read those documents
```

```
# ENCRYPTED PASSWORDS ARE THE DEFAULT FOR LINUX, WINDOWS 95 (WITH OSR2),
# WINDOWS 98/ME, WINDOWS NT 4 WITH SP3 OR HIGHER, WINDOWS 2000/XP/2003
  encrypt passwords = yes
  smb passwd file = /etc/samba/smbpasswd

# The following are needed to allow password changing from Windows to
# update the Linux system password also.
# NOTE: Use these with 'encrypt passwords' and 'smb passwd file' above.
# NOTE2: You do NOT need these to allow workstations to change only
#        the encrypted SMB passwords. They allow the Unix password
#        to be kept in sync with the SMB password.
  unix password sync = Yes
  passwd program = /usr/bin/passwd %u
  passwd chat = *New*password* %n\n *Retype*new*password* %n\n
*passwd:*all*authentication*tokens*updated*successfully*

# You can use PAM's password change control flag for Samba. If
# enabled, then PAM will be used for password changes when requested
# by an SMB client instead of the program listed in passwd program.
# It should be possible to enable this without changing your passwd
# chat parameter for most setups.
  pam password change = yes

# Unix users can map to different SMB User names
# SUPPORTS DIFFERENT WINDOWS AND LINUX USER NAMES
  username map = /etc/samba/smbusers

# This parameter will control whether or not Samba should obey PAM's
# account and session management directives. The default behavior is
# to use PAM for clear text authentication only and to ignore any
# account or session management. Note that Samba always ignores PAM
# for authentication in the case of encrypt passwords = yes
  obey pam restrictions = yes

# Most people will find that this option gives better performance.
# See speed.txt and the manual pages for details
  socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192

# Configure remote browse list synchronisation here
#  request announcement to, or browse list sync from:
# a specific host or from / to a whole subnet (see below)
;   remote browse sync = 192.168.3.25 192.168.5.255
# Cause this host to announce itself to local subnets here
;   remote announce = 192.168.1.255 192.168.2.44
# THIS SYCHRONIZES THE BROWSE LIST FROM THE LOCAL COMPUTER WITH THE WINS SERVER
# ON IP ADDRESS 172.22.30.13
remote announce = 172.22.30.13

# Browser Control Options:
# set local master to no if you don't want Samba to become a master
# browser on your network. Otherwise the normal election rules apply
# THIS ASSUMES YOU HAVE A DIFFERENT BROWSE MASTER SUCH AS THE PDC
  local master = no

# WINS Server - Tells the NMBD components of Samba to be a WINS Client
# Note: Samba can be either a WINS Server, or a WINS Client, but NOT both
# THIS POINTS TO THE IP ADDRESS OF THE WINS SERVER
  wins server = 172.22.30.4
```

```
#============================ Share Definitions ==============================
# THIS IS THE STANDARD SHARE FOR HOME DIRECTORIES ON THE LOCAL COMPUTER
[homes]
  comment = Home Directories
  browseable = no
  writeable = yes
  valid users = %S
  create mode = 0664
  directory mode = 0775
# If you want users samba doesn't recognize to be mapped to a guest user
; map to guest = bad user

# THIS WORKS FOR CUPS OR LPD PRINTERS
[printers]
  comment = All Printers
  path = /var/spool/samba
  browseable = no
  writable = no
# Set public = yes to allow user 'guest account' to print
  printable = yes

# THIS SETS UP A SHARE FOR TEMPORARY FILES / DOWNLAODS
[tmp]
  comment = Temporary Files
  path = /tmp
  read only = no
  guest ok = yes

# YOU CAH SET UP A SHARE FOR A GROUP OF USERS; THIS SHARE WORKS FOR THE
# SEATENGR GROUP AND USER mj
[shared]
  comment = Shared files
  path = /home/shared
  writeable = yes
  valid users = mj,@seatengr

# THIS IS ANOTHER SHARE FOR THE USERS elizabeth AND mj
[project]
  comment = Project Files
  path = /home/project
  writeable = yes
  valid users = elizabeth,mj
```

Index

Note that you can download the PDF file for this book from **www.hentzenwerke.com** (see the section "How to download files" at the beginning of this book). The PDF is completely searchable and will provide additional keyword lookup capabilities not practical in an index.

Special Characters

* (asterisk)
 deactivating users, 110
 with **rpm** command, 51
 in Samba configuration file, 116
\ (backslash character), continuation of commands, 147
>> (double forward directional arrow), with quotation marks, 147
% character, mounting directories, 143
%m variable, 87
. (period), hidden directories, 143
(pound sign), with **logrotate** script, 245
; (semicolon)
 activating tags, 95
 for comments, 84
' (single quotation marks), in .bash profile, 147
$SHELL value, 247
$USER variable, 147
32-bit Microsoft Windows operating systems, 59
4Suite package, 200
5, in commands, 254
5G command, 242

A

a command (vi), 242
A command (vi), 242
a2ps package, 200
Account Info settings, 109
ACL (Access Control Lists), 79, 123, 287
Active Directory support, in Samba 3.0, 287
adapter, 158
add group script command, 289
add machine script command, 289
add user script command, 113–114, 289
add user to group script command, 289
Administration Tools package group, 37, 38
administrative account, 105–106. *See also* root password
allowable drives, 22
Amanda (Advanced Maryland Automatic Network Disk Archiver), 39
anacron service script, 53
Apache Web server, 39, 57
apmd service script, 53
asterisk (*)
 deactivating users, 110
 with **rpm** command, 51
 in Samba configuration file, 116
at command, 247
atd service script, 53
auth methods variable, 289
authentication
 configuring, 30, 33–34
 definition, 29
 Microsoft user database, 114
 Samba, 34, 71–72, 287
Authentication Server, 72
Authoring and Publishing package group, 37
autofs service script, 53
autorun command, 49

B

backslash (\) character, continuation of commands, 146, 147
Backup Domain Controllers (BDCs), 88, 262
bad blocks, 23
"balanced trees" concept, 11
bash shell, 145
.bash_profile configuration file, 146
BDC (Backup Domain Controller), 88, 262
Berkeley Standard Distribution (BSD), 87
/bin directory, 10
bind (Berkeley Internet Name Domain), 39
bind interfaces only variable, 81
BIOS clock, 30
Block grace period, 132
/boot directory
 dual-boot configuration, 14
 function, 10
 mounted separately, 23
boot floppy, 24, 27, 31
boot loader, 15–16, 25–26, 40
boot process, 127–128, 250–256
broadcast address, 90–91
browse list, synchronizing, 90–91

browse master elections, 91–92
browse options, SWAT, 77–78

C

caching-nameserver, 39
cameras, information resource, 6
case sensitivity
 file names, 12
 passwords, 88
 Samba, 93, 229
cat command, 236
cd command, 232–233
central storage, 1
chkconfig command, 57
chmod command, 129, 249
chown command, 123
CIDR (Classless Inter Domain Routing), 90
CIFS (Common Internet File System), 60
CIPE (Crypto IP Encapsulation), 39
Classless Inter Domain Routing (CIDR), 90
client, 158
Client/Server Password Management submenu, 84
"command not found" error, 19
command-line interface
 accessing, 44
 advantages, 35
 checking settings, 261
 configuring Samba users, 112
 documentation, 52
 managing users and accounts, 106
 viewing networks, 62
commands. *See specific command names*
comment command, 99
comment variable, 74, 94
comments
 identifying network share, 96
 in Linux configuration files, 84
common directory, 98
Common Internet File System. *See* CIFS
computer accounts, configuring, 113–114
computer names, 61
cp command, 237, 238
crackers, 26, 39, 96
cron daemon, 247
cron scheduler, 243–245
crond service script, 54
CUPS (Common Unix Print System)
 configuration files, 201
 configuring printers with, 206–211
 installing, 199–201
 versions, 48
cups package, 200
cups service script, 54
cups system, 87
cw command (vi), 242

D

daemons, 2
DARPA (U.S. Defense Advanced Research Projects Agency), 11
data corruption, 12
date and time settings, 41
dd command (vi), 242
deactivating services, 56–57
deactivating users, 110
Debian Linux, 4
dedicated printer, 100–101
default case variable, 93
delete user from group script command, 289
/dev directory, 10
Development Tools package group, 37
DHCP (Dynamic Host Configuration Protocol), 29, 33, 169–171
directories. *See also* files; partitions; shared directories
 by client, 101–102
 common directory, 98
 copying all files in, 248–249
 file systems mounted on, 256
 group directory, 98–99
 hidden, 143
 home directory, 10, 14, 107
 mounting, 135, 139–142, 255–256
 organizing, 9–11
 permissions, 95
 private, 101
 structure, 230–232
disabling
 encryption, 157
 firewalls, 31
 signatures, 186
Disk Druid tool
 adding or editing partitions, 21–23
 alternative to FDISK.EXE, 18
DNS (Domain Name Service), 29
DNS Name Server package group, 37
dns proxy, 78
DNS Settings, TCP/IP Settings dialog, 178
domain
 connecting to, 137, 159–161, 171–172, 179, 188
 joining, 180
 re-entering user names and passwords, 194

viewing, 140
domain authentication, 71
domain logons command, 119
Domain Master Browser, 59
domain master command, 119
Domain member server, 295–298
domain network logons, 95–96
Domain share, 165–166, 173, 191
domain users, PDCs, 114–120
double forward directional arrow (>>), with quotation marks, 147
downloading
hardware information, 6–7
Red Hat Linux 9, 3
driver floppies, 27
drivers, finding, 7–8
dual-boot configuration, 14, 251

E

echo command, copying information, 147
editing
partitions, 21–23
print queue, 205–206
Editors package group, 36, 49
edquota command, 127, 132–133
empty space, after adding partitions, 22
encrypt passwords command, 119
encrypt passwords variable, 82, 88, 92
encrypted passwords, 30, 33, 72, 88, 106, 288–289
encryption, Windows 95, 92, 156–157
Engineering and Scientific package group, 36
error messages
"Access is denied" error, 193
"Command not found" error, 19
smbmount command, 249
"Subsys locked" error, 137
Esc command (vi), 242
/etc directory, function, 10
/etc/crontab command columns, 245
/etc/fstab command columns, 256
/etc/fstab configuration files, 135
/etc/group directory, 110
/etc/gshadow directory, 110
/etc/inittab directory, 253–255
/etc/passwd directory, 110
/etc/samba directory, 67
/etc/shadow directory, 110, 260–261
ext2 filesystem format, 11, 12
ext3 filesystem format, 11, 12
extended partition, 8

F

fdisk command, options, 19–21
fdisk, Linux version, 17
FDISK.EXE command, 18–21
Fedora Linux, 199
Fedora Rawhide, 40, 48, 287
File and Print server, configuring, 59
file management commands, 235–241
file names
case sensitivity, 12
in Samba 3.0, 287
file permissions, 95, 121–122
file servers. *See also* server file system; Web servers
basic functions, 1
File and Print server, 59
hardware considerations, 1
Kerberos 5, 289
samba-2* RPM server package, 136
setting up domain users
Microsoft user database, 114–118
Samba PDC database, 118–120
setting up users
considerations, 105–106
file and directory management, 121–123
Linux groups, 123–127
Linux user and group accounts, 106–111
quotas, 127–133
Samba users, 111–114
SMB, 34
WINS, 111
file sharing
passwords, 167
for workgroups, 1
file system type, 22
files. *See also* directories; log files
adding information to end of, 241
copying, 248–249
creating and deleting, 237–241
hidden, 144
installation, 248–249
organizing, 9–11
reading, 236–237
filesystem format
selecting, 11–15
terminology, 18
Filesystem Hierarchy Standard (FHS), 4, 8, 9–11
Filesystem, verifying, 132
filesystems, mounted on directories, 256
find command, 235–236
Finger utility, 39
FIPS.EXE command, 15–17

firewalls
configuring, 26, 31–33, 178
default services, 32–33
definition, 29
Samba servers, 66, 68–70
security levels, 31–32
FireWire (IEEE1394), information resource, 6
First Boot process, 29, 40–43
First Interactive Partition Splitter, 15–16
firstboot command, 43
firstboot service script, 54
font utilities, 200
FQDN (fully qualified domain name), 78
fragmentation, Microsoft Windows, 14
Free Software Foundation, GNU (GNU's Not Unix), 2
FTP (File Transfer Protocol), 29, 33
FTP server, 28
FTP Server package group, 37
functions service script, 54

G

G command (vi), 242
Games and Entertainment package group, 37
gateways, 32
getent passwd command, 117
ghostscript package, 200
Globals menu, SWAT, 76–78
GNOME Desktop Environment
as login terminal, 255
package group, 36, 38
text editor, 142, 144, 145
GNOME (GNU Network Object Model Environment), 43–44
GNOME Software Development package group, 37
.gnome2 directory, 143
GNU/Linux operating system, 2
gpm service script, 54
grace periods, quotas, 132
Graphical Internet package group, 36, 38
Graphics package group, 37, 39
Greenwich Mean Time, 30
group configuration files, 110
group directory, 98–99
Groups settings, 109
GRUB boot loader, 15–16, 25–26, 40, 250–252
Guest Account, 72
guest account variable, 95
GUI (graphical user interface)
Linux login batch files, 143–144
logging in at, 254–255
managing services from, 64–65
modules, 2
opening smb.conf file, 85

H

halt service script, 54
hard disk space. *See also* partitions
configuring, 19–20
minimum requirements, 5
hash mark (#), for comments, 84
head command, 237
hex codes, Linux partition types, 19
hidden files, 144
hide special files variable, 289
hide unwritable files variable, 289
high-security firewall, 31
"high-speed" connections, 26
/home directory, 10, 11, 13–14
Home menu, SWAT, 76
HOME variable, 244
hosts allow variable, 70, 77, 78
hosts deny variable, 70, 77, 78
HOWTO documents, Linux Documentation Project, 149
hpijs package, 200
HTTP (HyperText Transfer Protocol), 29, 33
HTTP server, 27
httpd service script, 54

I

i command (vi), 243
IDE hard disk, partitions, 8
idmap gid variable, 288
idmap uid variable, 288
info grub command, 250
init 5 command, 43
/initrd directory, 10
installation source, 249–250
interfaces variable, 81
InterNetNews, 39
IP addresses
in Samba configuration file, 116
as wildcards, 86
IP Settings, TCP/IP Settings dialog, 178
iptables RPM, 68, 70
iptables service script, 54
irda service script, 54
isdn service script, 54

K

kdcrotate service script, 54
KDE Desktop Environment package group, 36, 37
Kerberos 5 server, 33, 289
Kernel Development package group, 37
kernels, size considerations, 14
keyboard, minimum requirements, 5
keytable service script, 54
kill-9 PID command, 64
killall service script, 54
kill-HUP PID command, 64
kudzu service script, 54

L

LABEL command, 255
LAN (Local Area Network)
 browse master, 91
 definition, 30
 Linux installation and, 44
laptop computers, information resource, 6
LDAP (Lightweight Directory Assistance Protocol)
 BDC (Backup Domain Controller) as, 262–263
 database of user names and passwords, 151
 definition, 33
 passwords, 149
less command, 237
/lib directory, 10
Lindows, availability, 4
Linux distributions, 2, 4. *See also* Red Hat Linux
Linux Documentation Project, 149
Linux extended filesystems, 11
Linux groups, 123–127
Linux installation. *See also* file servers; software installation
 authentication, 33–34
 checking for bad blocks, 23
 coexisting with Microsoft Windows, 12, 14
 documentation, 42
 filesystem format, 11–15
 hardware
 checklist, 6–7
 considerations, 4
 drivers, 7–8
 minimum requirements, 5–6
 local, 24–26
 network, 26–28
 partitions, 4, 8–9, 13–15
 remaining steps, 30–31
Linux permissions, 121–122
linux rescue command prompt, 259–260
Linux workstations
 configuring Samba client packages, 136
 connecting to a domain, 137
 considerations, 135–136
 finding shared directories, 137–139
 Linux login batch files, 143–147
 Linux PDC and Linux Domain member server, 151–152
 on a Microsoft Windows Domain, 135
 mounting directories, 139–142
 peer-to-peer Workgroup, 147–149
 setting up accounts, 149–150
 Windows PDC and Linux Domain server, 150–151
lmhosts file, 67, 84
load printers command, 214
local master command, 119
locate command, 235, 236, 246–247
locking user accounts, 260–261
log files
 primary, 196–197
 rotated, 245
 workstation, 197
logical extent (LE), 18
logical partition, 8
logical volume (LV), 18
login name, 107
login scripts, 145
login shell, 107
logon path command, 119
logon path variable, 120, 155
logon scripts, 119–120, 154–155
logon script variable, 92
logrotate script, 245, 246
lokkit command, 68–70
/lost+found directory, 10
LPD (Line Print Daemon)
 alternatives, 201
 installing, 199
lprng system, 87
ls command, 233–235
LVM (Logical Volume Management), 9, 13, 18, 24

M

MAILTO command, 244
Make RAID Device dialog, 23
Mandrake Linux, availability, 4

map acl inherit variable, 289
MDS passwords, 33
medium-security firewall, 31
memory (RAM), minimum requirements, 5
Microsoft user database, 114–118
Microsoft Windows
 CIFS (Common Internet File System), 60
 coexisting with Linux, 14
 dual-boot option, 15
 Network Neighborhood, 61, 86
 NTFS partition and, 12
 RAWWRITEWIN.EXE utility, 24
 user categories, 181
 Windows XP Home Edition, limitations, 186
Microsoft Windows Domains, running Linux workstations, 135
Microsoft workstation. *See* Windows workstations
 Windows XP Professional
 configuring, 186–192
migration, in Samba 3.0, 288
/misc directory, 10
mkdir command, 237, 240
/mnt directory, 10
mobile devices, information resource, 7
modems, information resource, 7
monitor, 5, 31
monitoring system processes, 65
more command, 237
mount command, 128
mount point, 18, 22, 174
mounting directories, 135, 139–142, 255–256. *See also* shared directories
mouse, minimum requirements, 5
mv command, 237, 239
My Network Places, 61–63, 86
MySQL, 39. *See also* Apache Web server

N

named service script, 54
Nautilus interface, 44, 139, 140–141
navigational commands, 230–235
net command, 193, 287, 288
NetBEUI (NetBIOS Extended User Interface), 60
netbios name command, 119
netbios name variable, 81
NetBIOS (Network Basic Input Output System), 60, 116
netfs service script, 54
NETLOGON share, 95
network cards, 7, 26
Network ID, 189
network interfaces, 90
Network Logon Service, 95
Network Neighborhood, 61, 86
network printers. *See* shared printers
network properties, Windows 2000 Professional, 178
Network Servers package group, 37
network services, 2, 39–40, 54, 179
News Server package group, 37
NFS (Network File System), 30, 61, 63
NFS server, installing Red Hat Linux, 27
nfs service script, 54
nfslock service script, 54
NIS (Network Information Service)
 configuring, 33
 database of user names and passwords, 151
 definition, 30, 39
 passwords, 149
 slave servers, 262
nscd service script, 54
nsswitch.conf file, 115
NT Boot Loader, 253
NTFS partition, 12, 15
NTP (Network Time Protocol), 41
ntpd service script, 54

O

o command (vi), 243
O command (vi), 243
Office/Productivity package group, 37, 38
only guest command, 102
openss RPM, 89
operating systems, 61
/opt directory, 10
os level command, 119
OSR2 passwords, 156–157

P

package groups, 35, 47
Package Group Selection window, 39–40
PAM (Pluggable Authentication Module), 90
Partition Magic, 15
partitions. *See also* directories
 adding or editing, 21–23
 checking, 19–21
 considerations, 4, 8
 dual-boot option, 15
 for installing Linux, 12–13
 Linux partition types, 19
 naming, 8–9

splitting, 15–17
strategies, 11
typical scenarios, 13–15
passdb backend variable, 289
passwd command, 89, 107
Password Info settings, 109
Password menu, SWAT, 76, 83–84, 89
password server variable, 82, 137
passwords
case sensitivity, 88
changing, 89
CUPS, 207
encrypted, 30, 33, 72, 88, 92, 106, 288–289
invalid, with Samba, 196
LDAP, 149
MDS passwords, 33
NIS, 149
peer-to-peer Workgroup, 167
policies, 260–262
root password, 30, 105
in Samba 3.0, 288–289
Samba users, 112–113
for sharing files, 167
SWAT, 83–84
Windows 95, 156–157
Windows 98, 166
Windows ME, 166–167
PATH command, 244
path variable, 96, 97, 102
pcguest user, 87
pcmcia service script, 54
PDC (Primary Domain Controller), 2, 88, 105, 287, 291–295
peer-to-peer Workgroup, 87, 147–149, 185
Pentium class CPU, minimum requirements, 5
percent (%) character, mounting directories, 143
period (.), hidden directories, 143
permissions
default, 122
directories, 95
file, 121–122
Linux, 121–122
physical extent (PE), 18
physical volume (PV), 18
PID (Process IDs), 64
Pluggable Authentication Module (PAM), 90
POP3 (Post Office Protocol), 30
portmap service script, 54
Postfix service, 39
postgreSQL, 39. *See also* Apache Web server
PostScript printer driver, 205
preferred master command, 119
Primary Domain Controller (PDC), 2, 88, 105, 287, 291–295
primary log files, 196–197
primary partition, 8, 15–16, 23
print queue, 205–206
printcap name command, 214
Printer System Switcher, 201–202
printers. *See also* CUPS; LPD; shared printers
browsing for, 220
configuring, 199, 201–206
dedicated, 100–101
information resource, 7
installed, 86
shared, 94–102, 138
switching, 201–202
Printers menu, SWAT, 76, 79–80
printing command, 214
Printing Support package group, 37, 38
private directories, 101
private groups, 108, 109, 123–124
/proc directory, 10
protocol, 158
ps aux command, 65
ps command, 65
PV (physical volume), 18
pwd command, 232
PXE (Pre-booot eXecution Environment), 39

Q

:q command (vi), 243
:q! command, 242, 243
quota RPM, 127
quotas
activating, 132–133
boot process, 127–128
configuration files, 128–129
configuring groups, 131
configuring users, 129–131
considerations, 127
grace periods, 132

R

RAID (Redundant Array of Independent or Inexpensive Disks), 9, 13, 18, 23
RAID 0, 23
RAID 1, 18, 23
RAID 5, 18, 23
RAID device, 18
RAID Options dialog, 23
random service script, 54
Raw Print Queue, 213
rawdevices service script, 54

RAWWRITEWIN.EXE utility, 24
reading files, 236–237
reading service scripts, 55
realm command, 289
Red Hat Documentation, 42
Red Hat Enterprise Linux, versions, 3
Red Hat Linux. *See also* Disk Druid tool; file servers; Linux installation; software installation
 administrative tools
 managing installation files, 248–250
 managing the Linux boot process, 250–256
 regular user tasks, 260–263
 rescue mode, 257–260
 boot loader, 15, 25–26
 boot process, 250–256
 configuring, 33
 default GUI, 43
 directories, 10–11
 distributions, 3
 "extra" software, 53
 First Boot process, 29, 40–43
 GNOME (GNU Network Object Model Environment), 43–44
 GUI login screen, 42–43
 installing, 24–28
 package groups, 35–37
 private groups, 108, 109, 123–124
 quota RPM, 127
 RAID versions supported, 18
 redhat-config-languages utility, 26
 rescue mode, 257–260
 runlevels, 57
 Server installation, 25
 special groups, 124–127
 starting, 253–255
 UID (user individual ID), 66
 upgrading, 25
Red Hat Linux 9
 downloading, 3
 installing, 48–50
 samba-2* RPM server package, 136
 text editor, 49
Red Hat Linux Update Agent, 45–47, 48
Red Hat Network (RHN)
 installation, 44–47
 registering with, 42, 44
 subscription, 3
Red Hat Package Manager (RPM), 4, 29, 48. *See also* Samba servers
Red Hat Printer Configuration utility, 202, 212, 225–227
Red Hat Samba Configurator, 70–75, 111–114, 125
Red Hat User Manager, 106–111
redhat-config-languages utility, 26, 30
redhat-config-printer package, 200
redhat-switch-printer command, 201
registry files
 disabling encryption, 157
 disabling signatures, 186
ReiserFS filesystem, 11
reiserfs filesystem format, installing Linux, 12
remote browse master, 90–91
renice command, 246
repetitive tasks, script, 243
repquota command, 127
rescue mode, 257–260
RESTORRB.EXE command, 16
Retrieval/Installation settings, 45
RHCE (Red Hat Certified Engineer), 229
RHN. *See* Red Hat Network
rhnsd service script, 54
rmdir command, 237, 239–240
roaming profiles
 advantages, 96–98, 120
 setting up, 161–162
 Windows 2000 Professional, 182
 Windows NT/2000/XP, 182
 Windows NT 4 Workstation, 172–173
 Windows XP Professional, 190–191
/root directory, 10
root password, 30, 105–106
root password, 207
root user, 11, 110
rootnotify command, 253
RPM. *See* Red Hat Package Manager
rpm command, 48, 50–52, 200
RSH (remote shell), 40
runlevels, 57, 253–255
run-parts command, 244

S

Samba. *See also* SWAT
 authentication, 34
 case sensitivity, 93
 client packages, 136
 configuration files, 46, 115–116
 configuring, 59, 63
 connecting to a Linux print server, 222–224
 documentation, 48
 file servers, 111–114

logs, 196–197
PDC database
Printer Configuration utility, 202
process management, 64
sample configuration files
Domain member server, 295–298
Primary Domain Controller (PDC), 291–295
syntax, 195–196
user accounts, 111–114
variables, 81–82
Samba 3.0
new features, 287–288
new variables in configuration file, 288–289
samba RPM package, 148
Samba servers
configuring
configuration files, 67–68
firewalls, 66, 68–70
getting Samba, 66–67
global options, 86
with Red Hat Samba Configurator, 70–75
smb.conf file, 85
peer-to-peer Workgroup, 87
shared directories and printers, 94–102
variables, 84–85
samba-2* RPM server package, 136
samba-client package, 136
samba-common RPM, 84
samba-common RPM package, 136
samba-swat RPM, 75
saslauthd service script, 54
/sbin directory, 10
scanners, information resource, 7
scripts
migration, 288
for repetitive tasks, 243
service scripts, 52–57
SCSI hard disk, partitions, 8
seatengr command, 142
seatengr password command, 142
secrets.tdb file, 67
Secure Socket Layer (SSL), 89
security command, 119
security issues. *See also* firewalls
encryption, 156–157
limiting access to networks, 86
NFS, 63
Samba, 68–70, 288
share-level security, 149
SWAT, 77
updates, 48
security variable, 71, 81, 87–88
semicolon (;)
activating tags, 95
for comments, 84
sendmail, 40
sendmail service script, 54
server authentication, 71
Server Configuration Tools package group, 37, 38
server file system
Samba Server Configuration tool, 59
server types, 80
user categories, 65–66
variables, 81–82
Server installation, Red Hat Linux, 25
server string command, 119
servers. *See* file servers
Service Configuration utility, 56–57, 64, 68, 74
service scripts, 53–57
shadow passwords, 33
share variable, 148
shared directories. *See also* mounting directories
browsing, 184
changing settings, 74
commands, 149
connecting to, 150, 154–155, 162–164, 183–184
defining, 94–102
files, 141
finding, 137–139, 138
first-time connection, 183, 191
mapping to, 184
mounting, 194–195
navigating to, 174
security variable, 71
setup files, 142
user access rights, 166
verifying connection to, 145
on Windows-based networks, 135
shared printers. *See also* CUPS; LPD
authorizing connections to, 226
configuring, 211–215
connecting to from workstations, 215–221
defining, 94–102
over a network, 214–215
packages, 199–201
in Samba 3.0, 288
share-level security, 149
Shares menu, SWAT, 76, 78–79
SHELL command, 246
$SHELL value, 247
SHELL variable, 243–244

shells
definition, 2, 229
navigational commands, 230–235
SID (Microsoft-style security identifier), 67, 288
signatures, disabling, 186
Silicon Graphics (SGI), 12
single quotation marks ('), in .bash profile, 147
single service script, 54
size considerations, partitions, 22
smb password file command, 119
SMB (Server Message Block) protocol, 30, 34. *See also* CIFS
smb service script, 54
smb status command, 137
SMB Workgroup, 34
smbadduser command, 90, 126, 140
smbclient command, 138, 147
smb.conf file, 67, 84, 85, 105, 116, 119
smbd processes, 65
smbmnt command, 142, 143
smbmount command
adding to bash profile, 151
adding to Sessions dialog, 150–151
connecting to, 148
customized, 145
error messages, 249
mounting shared directories, 139
peer-to-peer Workgroup, 147
smbpasswd command, 113, 262, 289
smbpasswd file, 67–68
smbpasswd username command, 118
smbpasswd-X computername command, 67
smbumount command, 142, 148
smbusers file, 68, 74, 113
SMTP (Simple Mail Transfer Protocol), 30, 33
snmpd service script, 55
snmptrapd service script, 55
software installation
adding or removing packages, 49
basic components, 35
network services, 39–40
options, 35–39
package management, 48–50
Rawhide, 40, 48
Red Hat Network (RHN), 44–47
rpm command command, 50–52
uninstalling unnecessary software, 52–57
Sound and Video package group, 37, 39
sound cards, information resource, 7
special groups, 124–127
splashimage command, 252
SQL Database Server package group, 37
SSH (Secure Shell), 30, 33, 40
sshd service script, 55
SSL. *See* Secure Socket Layer
startup commands, 143
Status menu, SWAT, 76, 82–83
subnet mask, 90
"subsys locked" error, 137
SuSE, availability, 4
swap filesystem format, installing Linux, 12
swap partition, 9
swap partition, dual-boot configuration, 14
SWAT (Samba Web Administration Tool). *See also* CUPS
activating, 75
advantages, 59
browse options, 77–78
documentation, 75
menu options, 76
passwords, 83–84
security levels, 77
security options, 77
switches, **rpm** command, 52
synchronizing user databases, 262–263
SysAdmin, Audit, Network, Security (SANS) Institute, 48
syslog service script, 55
System Commander, 15
system crashes, 65
System Tools package group, 37, 38

T

tail command, 237
TCP/IP (Transfer Control Protocol/Internet Protocol)
configuring, 169–171
definition, 30
ports for configuring firewalls, 70
Settings dialog, 178
Telnet, 33, 40
testparm command, 195–196
text editor
GNOME, 142, 144, 145
reading service scripts, 55
selecting, 49
vi, 241–243
text login batch setup, 144–147
Text-based Internet package group, 37
text-mode interface, installing Linux, 27
text-mode network commands, 193–195, 231–232
time zone, 30
title command, 253
/tmp directory, 10

/ (top level) directory, 10, 14
touch command, 129, 237, 238
troubleshooting
 Samba, 195–197
 software installation, 51
trust relationships, in Samba 3.0, 288
trusted devices, 32
ttfprint package, 200
Turbolinux, availability, 4
Tux server, 40

U

UID (user individual ID), 108
uninstalling unnecessary software, 52–57
United Linux, 3
unix password sync command, 119
Update Agent. *See* Red Hat Linux Update Agent
updatedb command, 246
UPS (uninterruptible power supply), 254
URI (Universal Resource Identifier), 209
USB device, information resource, 7
Usenet, InterNetNews, 39
user authentication, 71
user categories
 Microsoft Windows, 181
 server file system, 65–66
user configuration files, 110
user databases, synchronizing, 262–263
User Data settings, 109
user groups, 130
user name/password database, in Samba 3.0, 288
user tasks, 260–263
useradd command, 87, 118
users, deactivating, 110
/usr directory, 10

V

/var directory, 10
vfat filesystem format, 12
vi alias 1 command, 241
video card, 5, 31
View menu, SWAT, 76, 83
VNC (Virtual Network Computing), 40
volume, 18
volume group (VG), 18
vsFTP (very secure FTP server), 40

W

wbinfo commands, 151
Web Server package group, 37
Web servers. *See also* file servers; firewalls
 Apache, 39
 firewalls and, 32
 Tux, 40
 VNC, 40
 vsFTP, 40
wildcards
 flexibility, 89
 IP addresses as, 86
 with **rpm** command, 51
winbind, 105, 114, 116–117
Winbind connection, 137
winbind gid variable, 137, 288
winbind service script, 55
winbind start command, 64
winbind status command, 137, 151
winbind uid variable, 288
Windbind architecture mapping, in Samba 3.0, 288
windbind uid variable, 137
Windows File Server package group, 37, 38
Windows workstations
 configuring connections
 considerations, 155–156
 connecting to a Domain share, Windows XP Professional, 191–192
 connecting to a Windows print server, 224–227
 connecting to a Workgroup share, Windows XP Professional, 192
 connecting to the Domain
 Windows 2000 Professional, 182–185
 creating a Domain share, Windows NT 4 Workstation, 176
 creating a Workgroup share, Windows NT 4 Workstation, 176–177
 preparing accounts, 153–155
 in Samba 3.0, 288
 setting up roaming profiles
 Windows 2000 Professional, 182
 text-mode network commands, 193–195
 troubleshooting
 Samba logs, 196–197
 Samba syntax, 195–196
 Windows 2000 Professional
 configuring, 177–185
 connecting to the Domain, 179–181
 connecting to Workgroups, 185
 Windows 95/98/ME
 connecting to the Domain, 156–167
 Windows 95/98/ME Workgroup share
 configuring, 167–168

Windows NT 4 Workstation
configuring, 168–177
configuring connections, 158–171
connecting to the Domain, 171–172, 173–175
connecting to Workgroups, 175
setting up roaming profiles, 172–173
Windows XP Professional
connecting to the Domain, 188–189
Winmodems, 7
Winprinters, 7
wins client variable, 137
WINS server, 111
wins server variable, 82, 262
WINS Settings, TCP/IP Settings dialog, 178
wins support command, 119
wins support variable, 82
WINS (Windows Internet Name Service) server
activating, 93
advantages, 59
configuring, 80
options, 78
Wizard menu, SWAT, 76, 80–82
workgroup command, 119
Workgroup share, 175, 185, 192
workgroup variable, 81, 86, 116
workgroupname variable, 148
workgroups, 1, 87, 147–149
workstation log files, 197

X

X Software Development package group, 37
X Window, 2, 35–36, 38
Xandros, availability, 4
Xfree86-font-utils, 200
XFS filesystem, 12
xfs filesystem format, installing Linux, 13
xfs service script, 55
xinetd service script, 55
XML tools, 200

Y

ypbind service script, 55